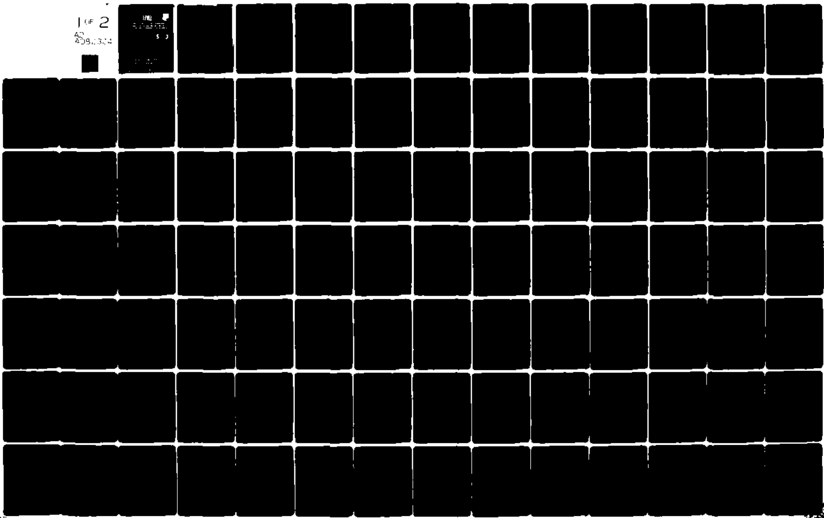
AD-A082 324 CONTROL DATA CORP MINNEAPOLIS MINN F/G 9/2
APPLICATIONS OF A RECONFIGURABLE ARRAY OF FLEXIBLE PROCESSORS I--ETC(U)
DEC 79 W R CYRE F30602-78-C-0065
UNCLASSIFIED RADC-TR-79-313 NL

1 OF 2
AD
A082324



RADC-TR-79-313
Final Technical Report
December 1979

LEVEL II

12

APPLICATIONS OF A RECONFIGURABLE ARRAY OF FLEXIBLE PROCESSORS IN INTELLIGENCE INFORMATION RETRIEVAL

Control Data Corporation

Walling R. Cyre

APPLICATIONS OF A RECONFIGURABLE ARRAY OF FLEXIBLE PROCESSORS

ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, New York 13441

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: RADC-TR-79-313	2. GOVT ACCESSION NO.
	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): APPLICATIONS OF A RECONFIGURABLE ARRAY OF FLEXIBLE PROCESSORS IN INTELLIGENCE INFORMATION RETRIEVAL	5. TYPE OF REPORT & PERIOD COVERED: Final Technical Report, Jan[redacted]78 — Apr[redacted]79
	6. PERFORMING ORG. REPORT NUMBER: N/A
7. AUTHOR(s): Walling R. Cyre	8. CONTRACT OR GRANT NUMBER(s): F30602-78-C-0065
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Control Data Corporation, 2800 East Old Shakopee Road, Minneapolis MN 55420	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 62702F, 45941607
11. CONTROLLING OFFICE NAME AND ADDRESS: Rome Air Development Center (IRDT), Griffiss AFB NY 13441	12. REPORT DATE: Dec[redacted]79
	13. NUMBER OF PAGES: ~~114~~ 119
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): Same	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE: N/A

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Same

18. SUPPLEMENTARY NOTES

RADC Project Engineer: John J. Maier (IRDT)

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Document Retrieval
Associative Processing
Processor Array

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The objective of this study was to investigate the applicability of a Reconfigurable Array of Flexible Processors, with enhanced word processing capabilities, to the problem of document retrieval.

The work performed during the study falls into three categories. First, the intelligence information retrieval application at AFSC Foreign Technology Division (FTD), Wright-Patterson AFB OH was analyzed for characteristics which might serve as system design and performance evaluation measures. The

DD FORM 1473, 1 JAN 73 — EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Ø16300

cont.

analysis was conducted by reviewing the data base and system usage characteristics at FTD and by synthesizing and analyzing a small experimental data base. From this, several areas in which the Reconfigurable Array with an associative processing unit might be used to advantage were identified, and a number of system sizing parameters were found.

The second category of work included the design modification and fabrication of an exploratory model of an associative unit, and its integration with a laboratory Reconfigurable Array. The exploratory model has a 4-bit slice architecture of which 32 cells are populated. Each associative cell has 1024 bits of storage and a full arithmetic-logic unit. The exploratory model is microprogrammable to permit connection with different types of processors as hosts, and to relieve the control load on the host processor.

The third category of effort included the development and exercising of a demonstration document retrieval system using a Reconfigurable Array of Flexible Processors, the Exploratory Development Model Associative Unit, and the experimental data base. The demonstration system capabilities cover the majority of search functions used at FTD, and served as a tool for both system diagnostics and for performance evaluation.

The results of the evaluations performed during the study indicate that a Reconfigurable Array with a fully populated Associative Unit should perform searches about 200 times faster than the current system.

Accession For

NTIS GRA&I ☒
DDC TAB ☐
Unannounced ☐
Justification ______

By ______

Distribution/

Availability Codes

Dist	Avail and/or special
A	

TABLE OF CONTENTS

TABLE OF CONTENTS (CONT'D)

LIST OF FIGURES

LIST OF FIGURES (CONT'D)

LIST OF TABLES

EVALUATION

The purpose of this effort was to demonstrate the feasibility of using an associative processor in conjunction with an array of microprogrammable processors for document retrieval. An experimental system was implemented and the retrieval of documents from a small data base was demonstrated. The results indicate that an array of processors with a fully populated associative unit should perform searches about 200 times faster than is achievable with current systems.

Such a system would be of extreme value in the intelligence community because of the very large data bases which must be searched. Also, the array of processors has been interfaced with a PDP-11 series computer. The AN/GYQ-21(V), which is the standard system for the intelligence community, is based upon PDP-11 computers.

Therefore, a full scale system, of the type demonstrated under this effort, could be utilized by intelligence agencies with minimal impact on existing hardware.

This effort is included in RADC TPO R1E, THRUST: INTEGRATED INTELLIGENCE SYSTEMS; SUBTHRUST: Intelligence Systems Concepts.

John J. Maier

JOHN J. MAIER
Project Engineer

1.0 INTRODUCTION

This report describes a study performed by the Information Sciences Division of Control Data Corporation for Rome Air Development Center under contract F30602-78-C-0065. The objective of the study was to investigate the applicability of a Reconfigurable Array of Flexible Processors, with enhanced word processing capabilities, to the problem of document retrieval for the assessment of foreign technology as performed by the AFSC Foreign Technology Division, Wright-Patterson AFB, Ohio.

The work performed during the study falls into three categories. First, the intelligence information retrieval application at FTD was analyzed for characteristics which might serve as system design and performance evaluation measures. The analysis was conducted by reviewing the data base and system usage characteristics at FTD and by synthesizing and analyzing a small experimental data base. From this, several areas in which the Reconfigurable Array with an associative processing unit might be used to advantage were identified, and a number of system sizing parameters were found.

The second category of work included the design modification and fabrication of an exploratory model of an associative unit, and its integration with a laboratory Reconfigurable Array. The exploratory model has a 4-bit slice architecture of which 32 cells are populated. Each associative cell has 1024 bits of storage and a full arithmetic-logic unit. The exploratory model is microprogrammable to permit connection with different types of processors as hosts, and to relieve the control load on the host processor.

The third category of effort included the development and exercising of a demonstration document retrieval system using a Reconfigurable Array of Flexible Processors, the Exploratory Development Model Associative Unit, and the experimental data base. The demonstration system capabilities cover the majority of search functions used at FTD, and served as a tool for both system diagnostics and for performance evaluation.

The results of the evaluations performed during the study indicate that a Reconfigurable Array with a fully populated Associative Unit should perform searches about 200 times faster than the current system.

2.0 THE DOCUMENT RETRIEVAL APPLICATION

This section presents the results of an analysis of the document retrieval problem related to intelligence activities. More specifically, the document retrieval system at AFSC Foreign Technology Division (FTD) was investigated to characterize system requirements. Necessarily, the scope of the investigation had to be limited, so the review of the FTD system was performed with respect to a basic set of search operations and a simplified data base structure. In addition, the data base analysis was restricted to the CIIO portion of the data base. In order to determine the growth pattern of the data base, a small experimental data base was also compiled and employed for the demonstrations described in Section 4.0. Where comparison between the CIIO data base and the experimental data base was possible, correspondence was found to be good.

2.1 DOCUMENT RETRIEVAL AT FTD

The AFSC Foreign Technology Division maintains a set of large data bases built on foreign literature in science and technology. A primary use of this system is to support intelligence analysts in the assessment of foreign technology. The analysis performed during this study was limited to the CIIO data base which contains references or documents which are two to five years old.

The retrieval mode investigated in this study was the search mode, wherein an intelligence analyst can form a search expression, a set of words or codes with logical connectives, and retrieve the identification numbers of all documents which satisfy that search expression. Some of the logical connectives or operators which may be used to formulate search statements (expressions) are listed in Figure 2-1 with indications of their relative frequencies of use during 1977. The OR operator in an expression such as "MICROWAVE OR RADAR" retrieves identifiers for documents containing either "MICROWAVE" or "RADAR". The AND operator returns identifiers for only those documents containing both terms. The ADJ operator requires that the two specified search terms appear adjacent in the documents, and in the specified order. The AND operation can be further restricted such that both terms must be in the same paragraph (PAR) or

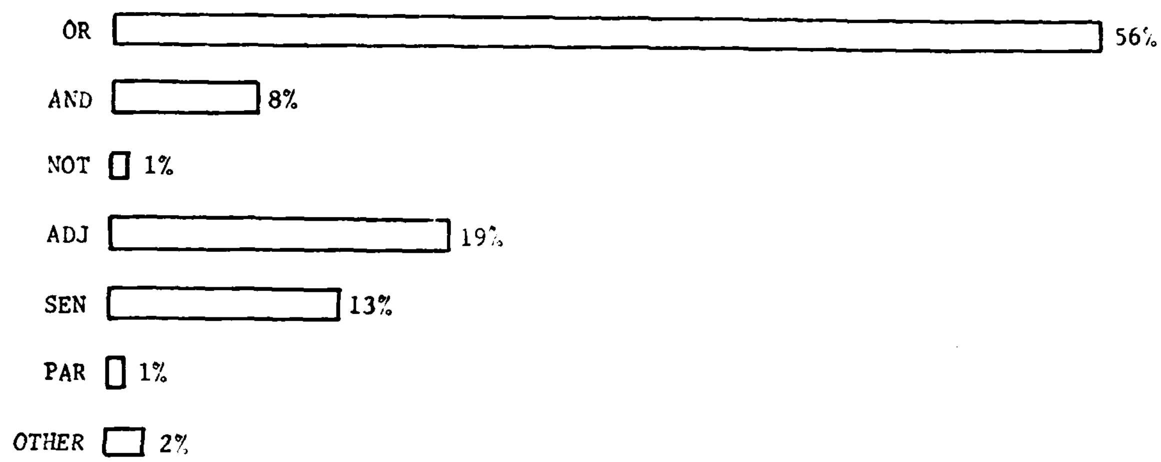


Figure 2-1. Search Operator Usage

the same sentence (SEN). The number of terms used to form a search expression is indicated in Figure 2-1. The average terms per search is 2.4 terms, implying an average of 1.4 connective operations per expression.

Another important and frequently used feature of the FTD system is the ability to use truncated search terms. For example, with the expression "RADAR AND TARGET$" the system finds the identifiers of documents containing any word beginning with "TARGET" before restricting the search to documents also containing the word "RADAR". The truncation can be bounded to a desired number of characters by specifying $n, where n is the number of characters. For example, "TARGET$2" includes "TARGETS" and "TARGETED", but excludes "TARGETING."

The performance of the FTD system is indicated in Figure 2-3. While the greatest fraction (22%) of searches is executed in less than two seconds, some searches require over 15 minutes for completion. The average execution time for a search was 35.8 seconds during 1979. With an average of 1.4 operations per expression, this implies that search operations are completed in an average of 25.6 seconds each. Later sections of this report describe a method by which a Reconfigurable Array of Flexible Processors and an Associative Memory can lead to dramatic improvements in the system performance.

In the following subsection, the structure of the data base and search procedures for document retrieval are examined in greater detail.

2.2 A DOCUMENT RETRIEVAL SYSTEM MODEL

The document retrieval system model described here is a highly simplified form of the FTD system, but exposes the fundamental system elements which affect performance. The model has two parts: a data base structure and a set of search procedures.

the same sentence (SEN). The number of terms used to form a search expression is indicated in Figure 2-1. The average terms per search is 2.4 terms, implying an average of 1.4 connective operations per expression.

Another important and frequently used feature of the FTD system is the ability to use truncated search terms. For example, with the expression "RADAR AND TARGET$" the system finds the identifiers of documents containing any word beginning with "TARGET" before restricting the search to documents also containing the word "RADAR". The truncation can be bounded to a desired number of characters by specifying $n, where n is the number of characters. For example, "TARGET$2" includes "TARGETS" and "TARGETED", but excludes "TARGETING."

The performance of the FTD system is indicated in Figure 2-3. While the greatest fraction (22%) of searches is executed in less than two seconds, some searches require over 15 minutes for completion. The average execution time for a search was 35.8 seconds during 1979. With an average of 1.4 operations per expression, this implies that search operations are completed in an average of 25.6 seconds each. Later sections of this report describe a method by which a Reconfigurable Array of Flexible Processors and an Associative Memory can lead to dramatic improvements in the system performance.

In the following subsection, the structure of the data base and search procedures for document retrieval are examined in greater detail.

2.2 A DOCUMENT RETRIEVAL SYSTEM MODEL

The document retrieval system model described here is a highly simplified form of the FTD system, but exposes the fundamental system elements which affect performance. The model has two parts: a data base structure and a set of search procedures.

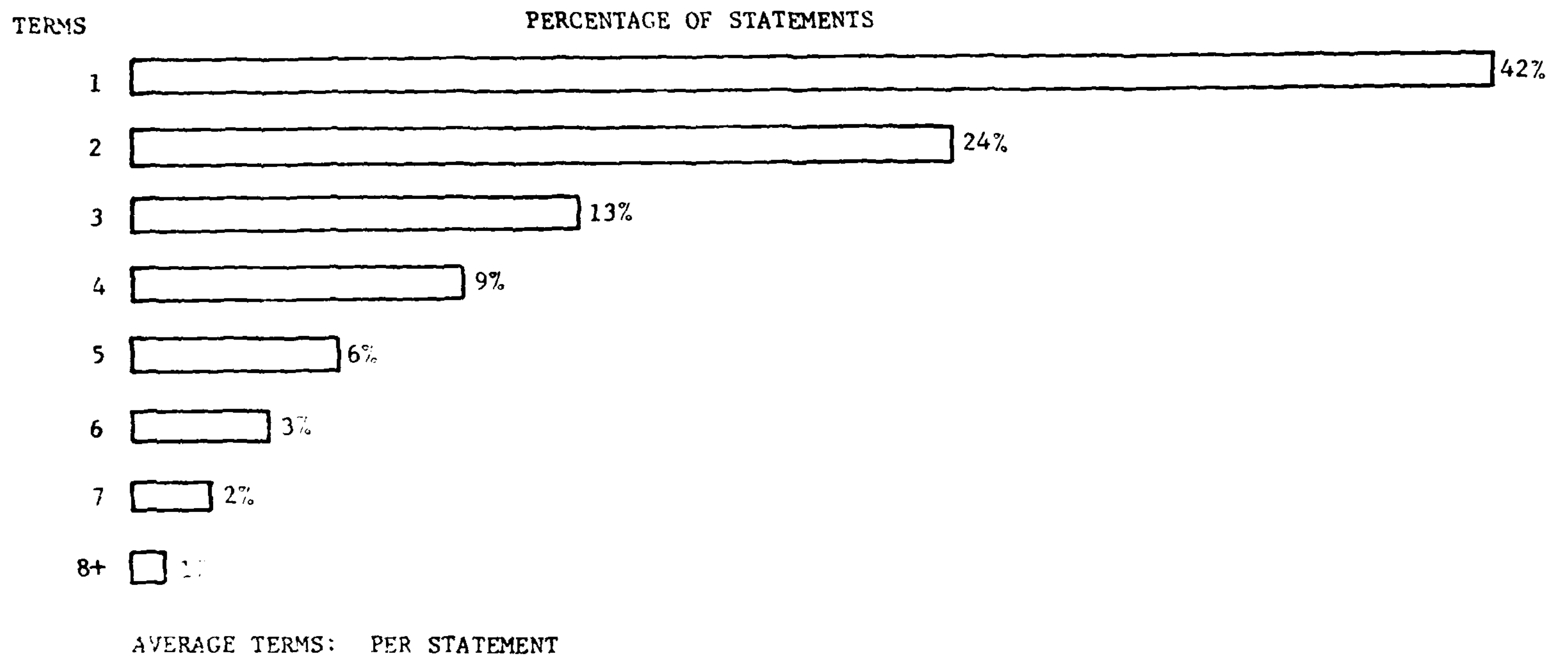


Figure 2-2. Terms Per Search Statement

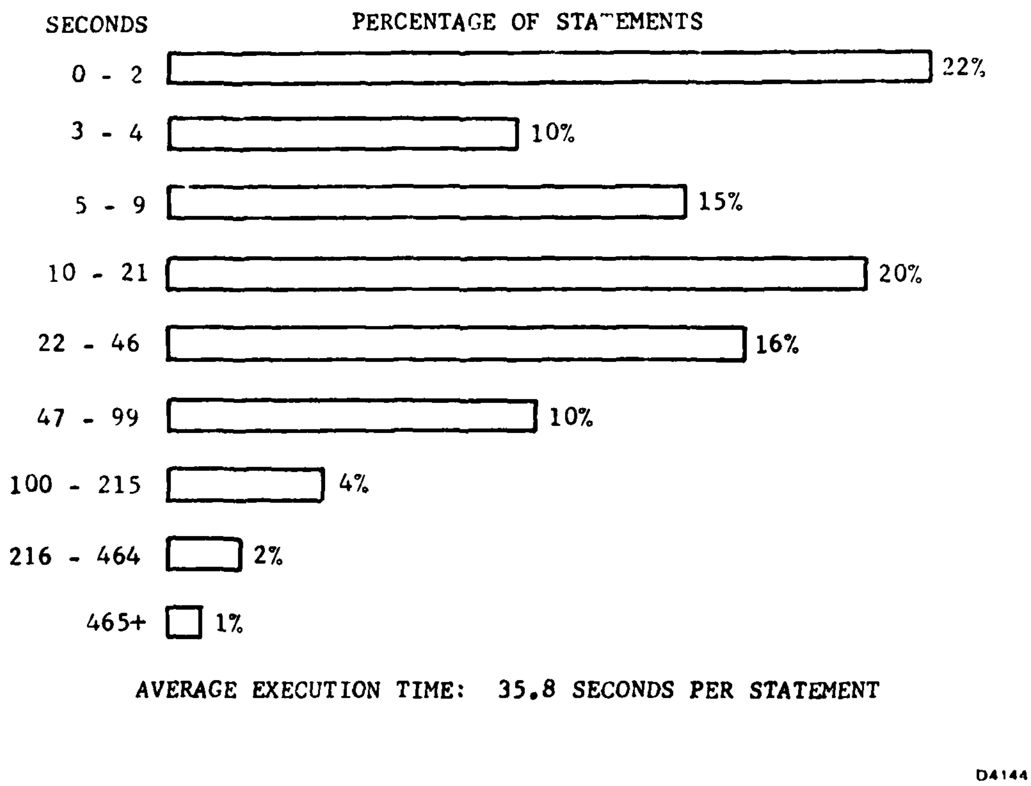


Figure 2-3. Search Statement Execution Time

2.2.1 A Data Base Structure

The foundation of the document retrieval data base is a set of documents. The set of documents is formulated into a TEXT file in which each document occupies a variable length record. Each document record has two fields, the first of which is the document identification number, and the second is the string of words of the document. Each document WORD is a character string delimited by blanks (spaces). Each word of a document can be identified by its document number (DN) and its word sequence number (WN). Thus, the two instances of the article A in the sentence fragment "A SHIP OR A BOAT" of the TEXT file are distinct and identifiable.

A second major file of the data base is the DICTIONARY file. Every record of the DICTIONARY contains a distinct character string or dictionary TERM which is represented at least once as a WORD in the TEXT file. Each DICTIONARY record has two other fields. One is the number of occurrences, NOC, of the TERM in the TEXT, and the second is an address for an OCCURRENCE LIST for the document number, word number pairs (DN, WN) identifying all occurrences of the TERM as a WORD in the TEXT file. The third major file is the OCCURRENCE FILE, whose records are the varying-length occurrence lists of the dictionary terms.

The general procedure for retrieving the documents containing a specified term is to look up the term in the DICTIONARY to find its occurrence list which in turn yields the desired document numbers. To facilitate the dictionary lookup operation, the dictionary is segmented or partioned into a mutually exclusive set of subfiles or segments. The fourth and final file of the data base is the dictionary SEGMENT TABLE which serves as a hash table to enter the dictionary. For current purposes, each segment table record consists of a KEY and the address of a dictionary segment. In the FTD system, the key is a pair of characters, and every possible pair is represented in the table. The table entries are stored by the character codes so the table address may be computed rather than looked up.

A summary of the data base files appears in Table 2-1. The access structure for the data base is unidirectional, via the SEGMENT TABLE, to the DICTIONARY, the OCCURRENCE file, and the TEXT file. This access structure is illustrated in Figure 2-4. The characteristics of the CIIO data base at FTD are interpreted with respect to this data base model in a later subsection of this report.

2.2.2 Search Procedures

A general procedure for document retrieval in the data base described in the previous subsection is illustrated in Figure 2-5. First, the search statement (search expression) must be decomposed or passed to identify the search terms and search operations. Next, the data base access structure is traced through the SEGMENT TABLE, DICTIONARY FILE, and OCCURRENCE FILE to retrieve an occurrence list for each search term. Note that a truncated search term will require retrieval of a set of occurrence lists. Then, the occurrence lists are operated on by the search operations to produce a result occurrence list, which may then be reduced to extract a list of identifiers for documents satisfying the search expression. The result occurrence list is necessary to allow the result of a search expression to be used as a pseudo search term for a subsequent search expression as is allowed in the FTD document retrieval system. In this general search procedure, the execution of the search operations imposes the greatest load on the computational resources, and warrants closer examination.

For the document retrieval model here, occurrence lists are viewed as sets of document number, word number pairs so that search operations can be formulated in set theoretic notions. Although the occurrence lists of the FTD system are ordered sets, this ordering restriction is not imposed in the model in order to preserve greater generality. The formulations of the search operations indicated in Figure 2-1 are given in Table 2-2. In the formulation of the search operations, AND, OR, and NOT directly correspond with union, intersection, and set difference respectively. Adjacency, ADJ, corresponds with set intersection, but the word numbers in the elements for

TABLE 2-1. FILES OF THE MODEL DATA BASE

FILE NAME	RECORD LENGTH	RECORD FIELDS		
		NAME	USE	EXAMPLE
TEXT	Varies	DN	Document Number	266
		DOCUMENT	Text	Side-Looking Air-Borne Radar: A Radar Antenna Mounted on . . .
DICTIONARY	Fixed	TERM	Character String	Radar
		NOC	Number of Occurrences	
		AD	Occurrence List Address	
OCCURRENCE	Varies	(DN,WN)	(Document Number, Word Number)	(266,3), (266,5)
SEGMENT TABLE	Fixed	KEY	Segmentation Key	RA
		SAD	Segment Address	

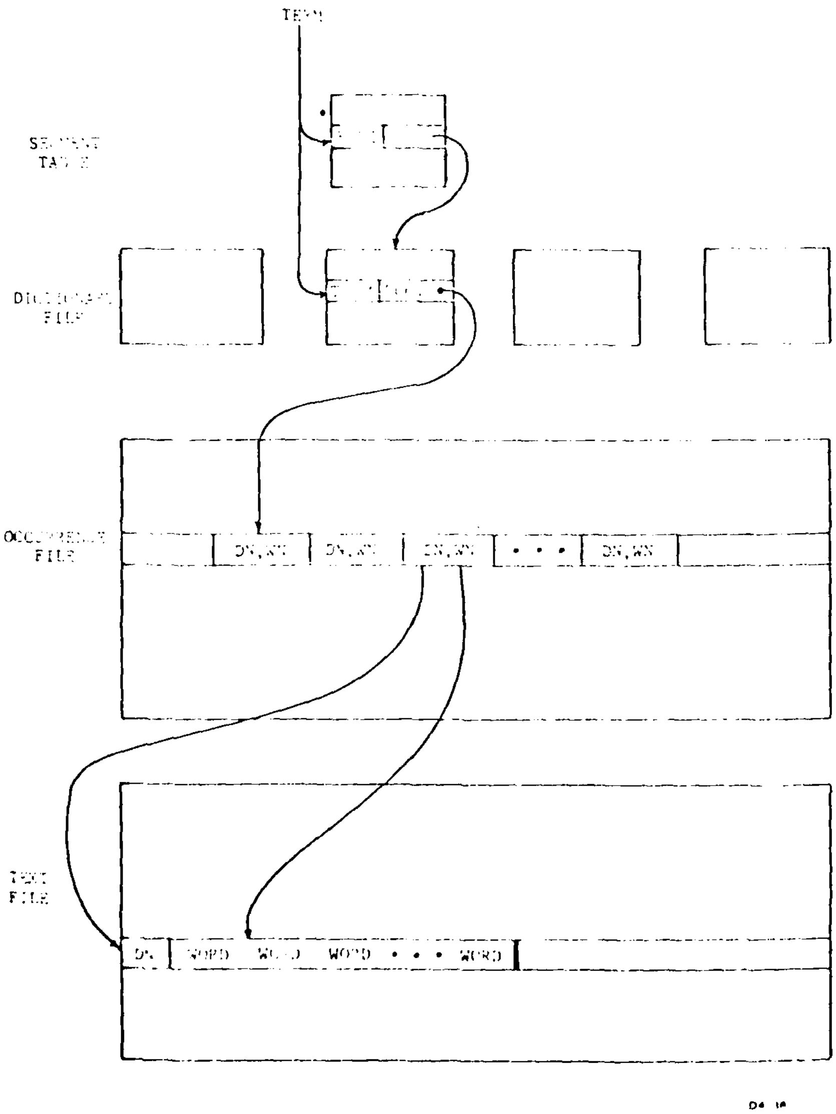


Figure 2-4. Data Base Access Structure

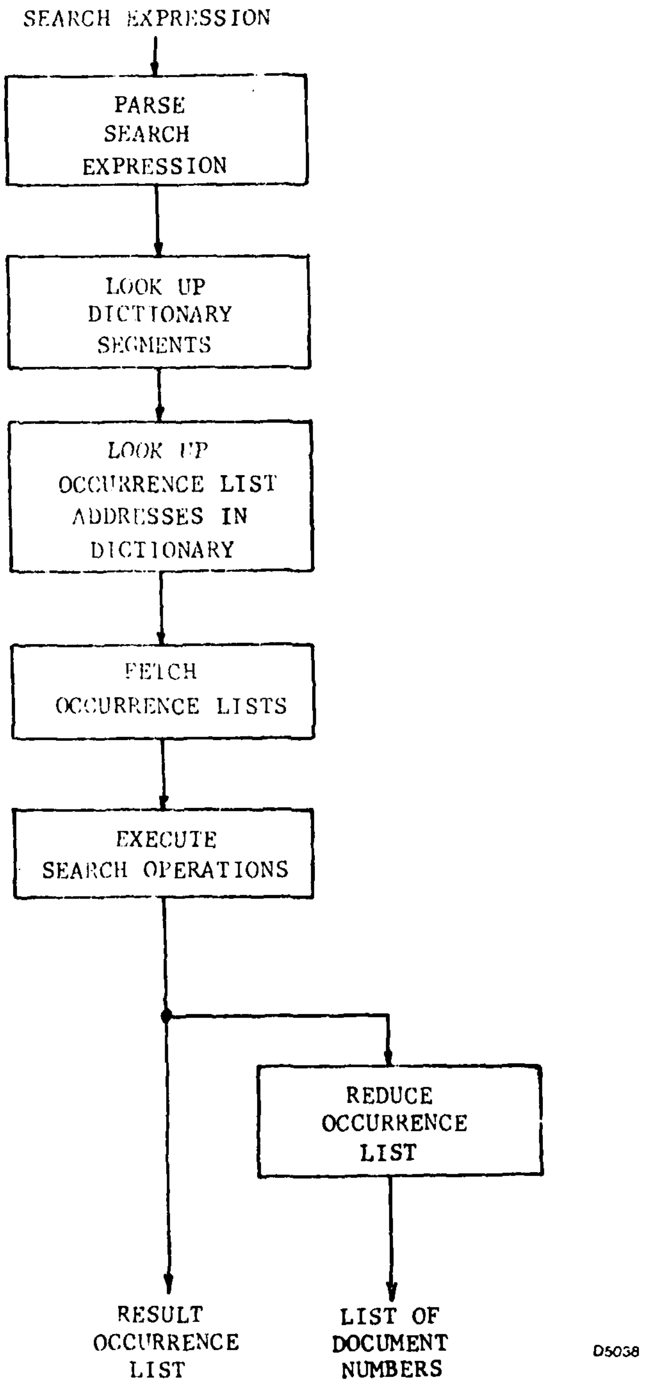


Figure 2-5. A General Search Procedure

TABLE 2-2. FORMULATION OF SEARCH OPERATIONS

SEARCH OPERATOR	SET OPERATION	DEFINITION OF RESULT
OR	Union	$C = \{(d,w) \mid (d,w) \in A \text{ or } (d,w) \in B\}$
AND	Intersection	$C = \{(d,w) \mid (d,w) \in A \text{ and } (d,w) \in B\}$
NOT	Difference	$C = \{(d,w) \mid (d,w) \in A \text{ or } (d,w) \notin B\}$
ADJ	Intersection	$C = \{(d,w) \mid (d,w+1) \in A \text{ and } (d,w) \in B\}$
SEN	Instersection	$C = \{(d,s) \mid (d,s) \in A \text{ and } (d,s) \in B\}$
PAR	Intersection	$C = \{(d,p) \mid (d,p) \in A \text{ and } (d,p) \in B\}$

A - occurrence list of left search term

B - occurrence list of right search term

C - result occurrence list

(d,w) - document number, word number pair

s - sentence number

p - paragraph number

the search term to the left of the operator must be incremented before intersecting the occurrence lists. Although the SEN and PAR are not supported by sentence and paragraph numbers of text words in the model data base, they could be treated as indicated in Table 2-2. With respect to the use of truncated terms as allowed by the FTD, the occurrence list corresponding to a truncated term is merely the cumulative union (OR) of occurrence lists for the dictionary terms included under the truncation.

In further examining the operations listed in Table 2-2, it may be noted that the set operations require only the detection of matching elements in the operand sets. For Union, the result occurrence list is one of the operand lists plus the second operand list with matches deleted from it. For intersection, the result consists of only the matching elements, and the difference (NOT) may be formed by deleting matching elements from the occurrence list for the left search term.

A procedure for executing these search operations or matchings depends on the nature of the available hardware. Two approaches are mentioned here, one of which employs an associative memory. These approaches are illustrated in Figure 2-6. For the conventional approach, the occurrence lists are first sorted into numerical order where (DN,WN) is treated as a single number. This step may be avoided if the occurrence lists are ordered sets as is the case at FTD. Before the sorting, the word numbers of the left occurrence list will have to be incremented if the ADJ operation has been specified. The second step of the conventional procedure involves the merge of the sorted files. During the merge, matches will be detected and only the derived elements need be saved for the result occurrence list. The result list will be sorted as a consequence of this approach so resorting will not be required if it is used in a subsequent operation.

The computational cost of the conventional approach is proportional to $(n_1 \log_2 n_1 + n_2 \log_2 n_2)$ where n_1 and n_2 are the numbers of elements in the occurrence lists. Most of this cost is due to the sort (Mergesort) which has a cost n_i proportional to $n_i \log_2 n_i - n_i$ for each list. The merge operation

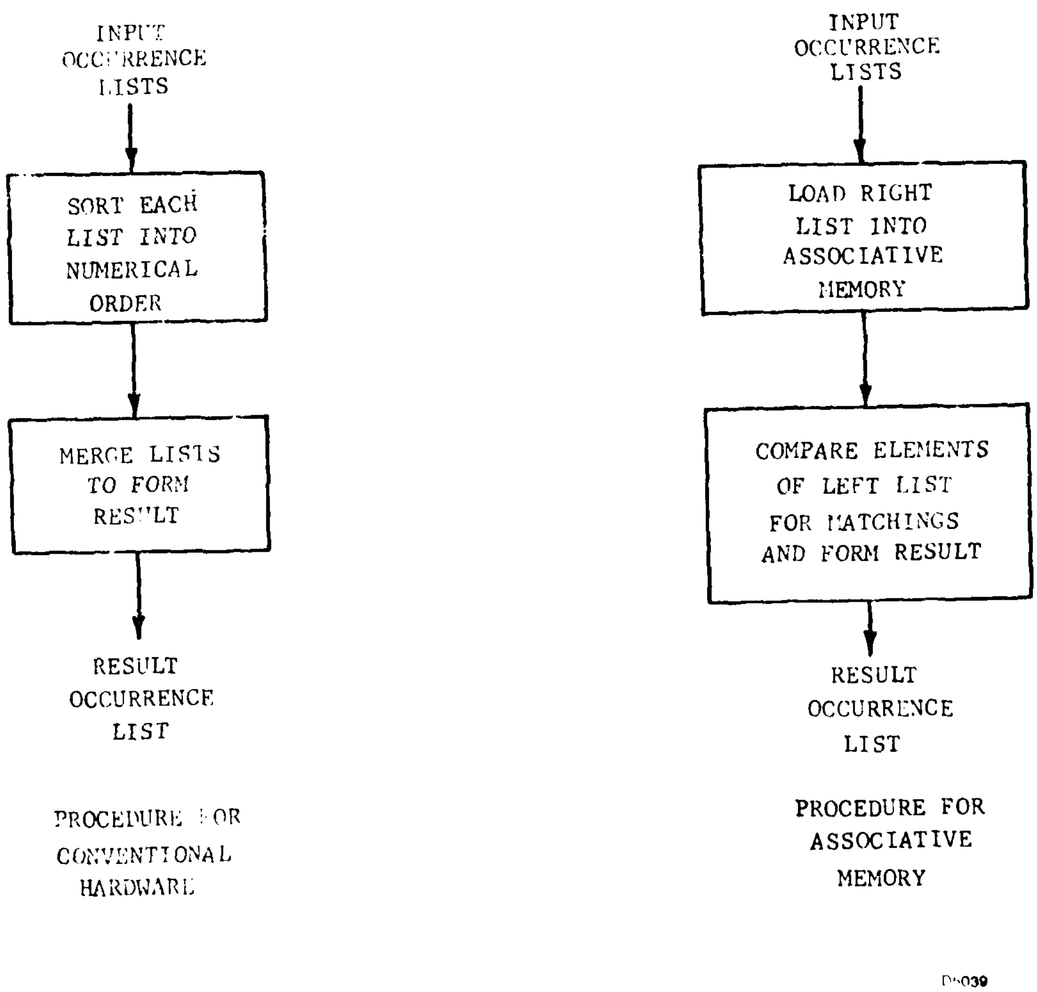


Figure 2-6. Two Procedures for Search Operations

is proportional to $n_1 + n_2$. The computational cost is strongly influenced by whether the occurrence lists are ordered or not. If the lists are unordered, the OR operation is trivial, since no distinct terms can occupy the same word position in the same document. Thus, the operand sets are mutually exclusive and no matching needs to be looked for. With ordered lists, the result must also be ordered so that the OR operation requires a merge of the operand lists to assure ordering in the result list.

To assume the existence of an Associative Memory to support the search operation, a different approach must be considered One property of an associative memory is that it allows a bit pattern to be compared with a set of bit patterns stored in the memory as a single operation. Further, the stored bit patterns need not be ordered. When such a comparison is made, the output of the memory indicates if any matches occurred. The associative procedure then requires that one occurrence list be stored in the associative memory. Then, each element of the other list is compared with the contents of the associative memory. Since ordering of the lists is not required, the OR operation is trivial, requiring only the concatenation of the occurrence lists. The computational cost for the other operations is proportional to the sum of elements of the operand lists n_1 and n_2, with n_2 operations to load the associative memory and n_1 comparisons to execute the search.

A summary of the computational cost of these procedures is presented in Table 2-3. As indicated, the associative memory offers the lowest total cost based on the relative frequencies of operation usage given in Figure 2-1. Note that the ADJ, SEN, and PAR search operations have been included with the AND operation because of their similarities. As will be seen in later sections, after greater hardware detail is considered, the performance improvement with associative memory is more dramatic than indicated by Table 2-3.

An associative memory unit offers advantages in other areas of the document retrieval problem. One area is the segment table look-up task. The FTD system uses the numeric code for the first two letters of a term as an

TABLE 2-3. COMPUTATIONAL COSTS OF SEARCH OPERATIONS

SEARCH OPERATION	FREQUENCY	CONVENTIONAL HARDWARE		ASSOCIATIVE HARDWARE
		Unordered	Ordered	
OR	.56	--	n	--
AND*	.40	n $\log_2 n$	n	n
NOT	.01	n $\log_2 n$	n	n
TOTAL COST	.97	.41 $n\log_2 n$	.97n	.41n

n total elements in both operand occurrence lists

* includes the ADJ, SEN, and PAP operations

address in the segment tabie. This not only results in a large table with vacant entries (such as for the letter pair QZ) but also imposes an uneven segmentation of the dictionary leading to uneven look-up times and complication in the management of the DICTIONARY file when new terms are added. By storing the segment table in an associative memory, any number of first characters can be used as the key. A comparison is made for the first (lexicographically) entry and its associated segment address pointer is read out. This allows all segments to have about the same size through choice of the individual keys. The number of dictionary segments is equal to the number of associative memory words.

Another use of an associative memory which should prove quite effective is in supporting dynamic relocation or paging of dictionary segments and occurrence lists among levels in a memory heirarchy. For example, the terms having the greatest current interest to intelligence analysts might be stored in a fast-access memory medium.

Before going into further detail on the mechanics of a document retrieval system, some of the characteristics of the data bases which affect the system design are explored in the following paragraph.

2.3 CIIO DATA BASE CHARACTERISTICS

FTD has several data bases to support its assessment of foreign technology and other intelligence activities. Of these, the CIIO data base founded on unclassified documents aquired two to five years ago was selected as an example document retrieval data base for this study.

The number of documents covered by the CIIO data base is limited by the system capacity, and, thus, is relatively constant. The data base is updated weekly, but twice a year a subset of documents is dropped to accommodate the updating. The CIIO data base characteristics described here were derived from a listing of the dictionary terms and corresponding numbers of occurrences (postings) for the CIIO data base as it was in September, 1977.

Two dictionary term lists for September, 1977 were supplied in microfische form by FTD. In one list, the terms were ordered according to their numbers of occurrences. The second list was lexicographically ordered. The lists showed that the dictionary had 2,113,651 terms of which 1,145,793 appeared only once as words in the document text (had single occurrences). The most frequently occurring term had 1,367,781 occurrences. The occurrence-ordered term list was sampled to obtain a distribution of occurrence list lengths, and from this the probability density function of Figure 2-7 was calculated. Figure 2-7 shows the probability, $P_r\{n\}$, that a randomly selected term of the CIIO dictionary has n occurrences in the document texts. (The dots indicate the sampling of the dictionary to obtain the occurrence list length distribution.) Equation 2-1 provides a simple expression (a straight line on the log - log plot) for the trend shown in Figure 2-7.

$$Pr\{n\} = 0.542 \times n^{-1.826} \qquad (2\text{-}1)$$

Based on Equation 2-1, the average number of occurrences is only 33.6 even though over a thousand terms have more than 10,000 occurrences. Equation 2-1 also predicts that the document text file contains over 71 million words. Also, since the occurrence file contains an entry for each document word, the occurrence file must allow 71 million entries.

2.4 CHARACTERISTICS OF FTD SEARCHES

The search operations on the occurrence lists of search terms represent a heavy computational load on the FTD system. This is not apparent if it is assumed that search terms used by the intelligence analysts are uniformly distributed over the dictionary (equivalent to randomly selected). Random selection indicates that the average occurrence list for search terms should have 33 entries and that 98% of the search terms should have fewer than 70 occurrences.

To investigate the occurrence list lengths of actual search terms, an audit for terminal activities for the CIIO data base was obtained from FTD

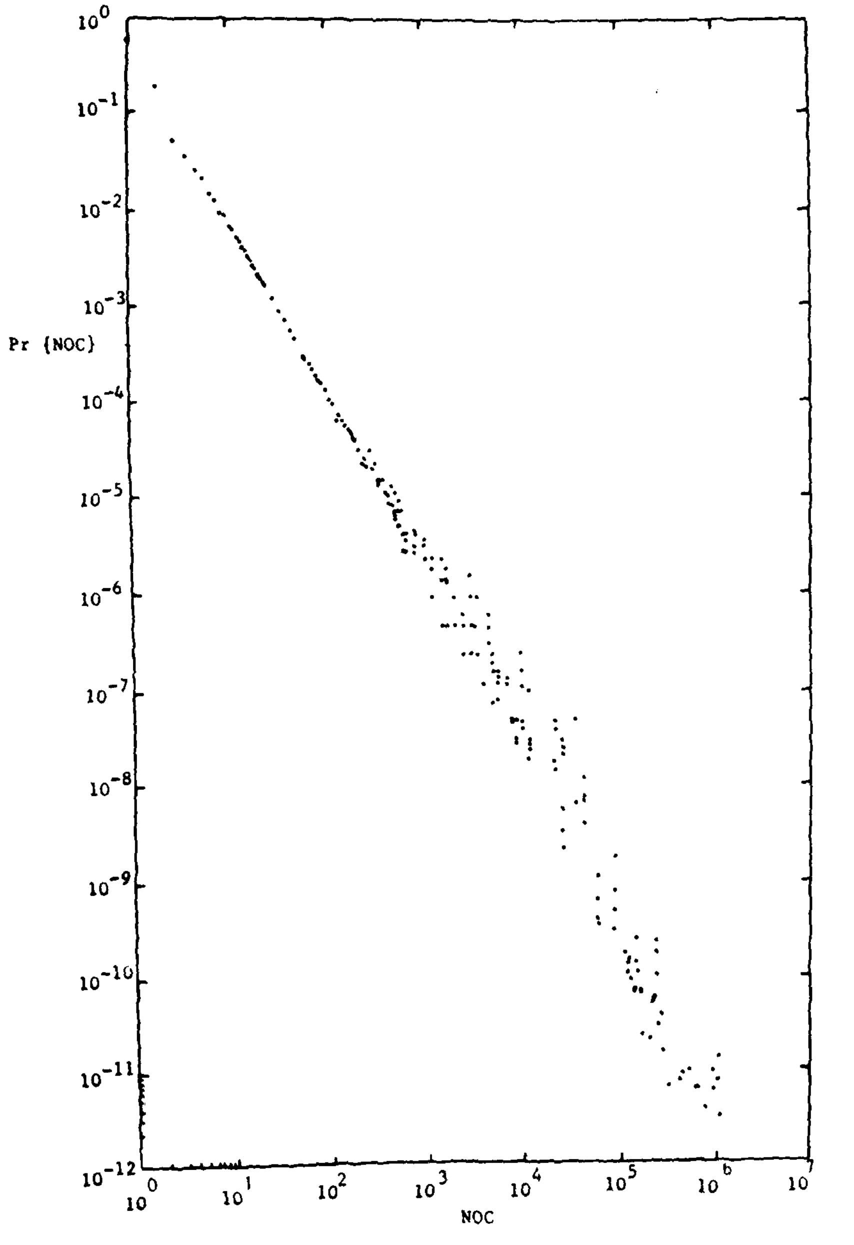


Figure 2-7. PDF of Number of Occurrences (CIIO Data Base)

on magnetic tape. This data was scanned to obtain a sample of 378 search terms used by analysts. Each of these terms was looked up manually in a lexicographically ordered dictionary term list to determine their occurrence list sizes. The search terms sampled included 138 truncated terms which covered 509 dictionary entries with an average of 3.7 dictionary terms per truncation. Of the 378 search terms, 57 had no dictionary entries.

The distribution of occurrence list lengths for actual search terms (including those covered by truncations) is illustrated by the probability density function of Figure 2-8. Comparison of Figures 2-7 and 2-8 shows that low occurrence dictionary terms are used relatively infrequently as search terms. The average number of occurrences per search term is 2905 occurrences per term, a substantial difference from the 33.6 occurrences per dictionary term. Further, only 64% of the search terms have fewer than 70 occurrences where 98% of the dictionary terms have fewer than 70 occurrences.

The distribution of occurrence list lengths for terms used by intelligence analysts indicates that the computational load due to search operations is much greater than might be expected from considering the dictionary characteristics alone. This difference in load is, in fact, nearly two orders of magnitude. For the associative memory approach, the implications are that the memory must be relatively large, and the advantages of the associative memory will be magnified.

2.5 AN EXPERIMENTAL DOCUMENT RETRIEVAL DATA BASE

An experimental data base was constructed during the work reported on here to serve as an object for studying data base characteristics and as the basis of a demonstration document retrieval system employing a reconfigurable array of processors. The small size of the experimental data base permitted many experiments to be performed on it, allowing close inspection of its growth characteristics.

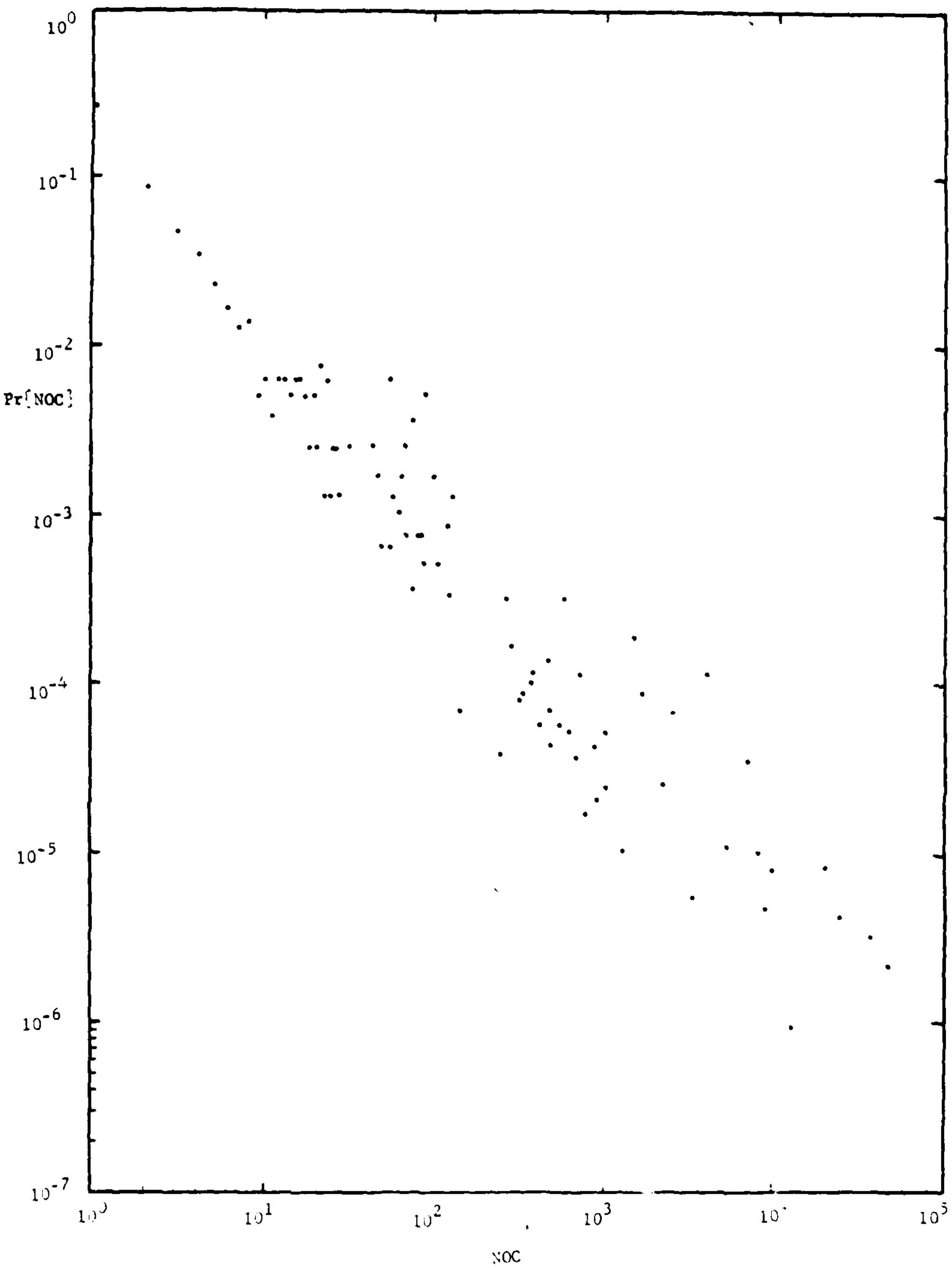


Figure 2-8. PDF of Number of Occurrences (Actual Search Terms)

2.5.1 The Experimental Data Base Documents

The documents of the FTD CIIO data base consist of unclassified scientific and technical literature from foreign sources. This implies broad coverage of many scientific and technical disciplines. In order that the experimental data base be comparable with the CIIO data base, the documents of the experimental data base were selected from Scientific American magazine, and so that many documents could be included, only extracts from the Scientific American articles were used. A document of the experimental data base, called the SA data base, consisted of the title, the publication date, the last names of the authors, and the 25 to 40-word summary on the first page of the printed article. For example, Figure 2-9 is a reproduction of page 84 from the October, 1977, issue of Scientific American (SA), and Figure 2-10 shows the extracted information, formatted as a document in the experimental data base. For the experimental SA data base, 192 documents from the January 1976 through December 1977 issues of Scientific American were used.

2.5.2 Organization of the Experimental SA Data Base

The structure used for the experimental data base is basically that shown in Figure 2-4. Some additional information was carried in the various files to support certain experiments. The records (documents) of the DOCUMENT file have the format shown in Figure 2-11. Document words are delimited by blanks, and the valid character set was restricted to the alpha-numerics, the hyphen, and the apostrophe. All other characters were considered to be blanks. As in the FTD system, the dictionary is segmented on the first two letters of the dictionary terms.

The record format for the DICTIONARY file is illustrated by Figure 2-12. Each dictionary term was assigned a unique identification number (WC). All terms were truncated to their first 16 characters, and because of the small size of the SA data base, no ambiguities were introduced. Each dictionary record includes counts of the number of occurrences (NOC) of the term and the number of characters in the term (NC). The final field of the record can have

Side-looking Airborne Radar

A radar antenna mounted on the belly of an aircraft and aimed to the side can record microwave images of terrain in striking detail, regardless of the weather or the time of day or night

by Homer Jensen, L. C. Graham, Leonard J. Porcello and Emmett N. Leith

The first aerial images of the earth's surface, made as an aid to topographic mapping and military reconnaissance, were crude photographs taken from balloons in the middle of the the microwave wavelengths of between one centimeter and 30 centimeters.

Radar systems provide their own illumination, and microwaves of certain frequencies are little affected by the at- spot. The image of each scan is retained by the cathode-ray tube so that a picture is continuously maintained.

Experience has shown that most of the images made by circular-scan radar

Figure 2-9. Reproduction of Page 84 from Scientific American, October 1977

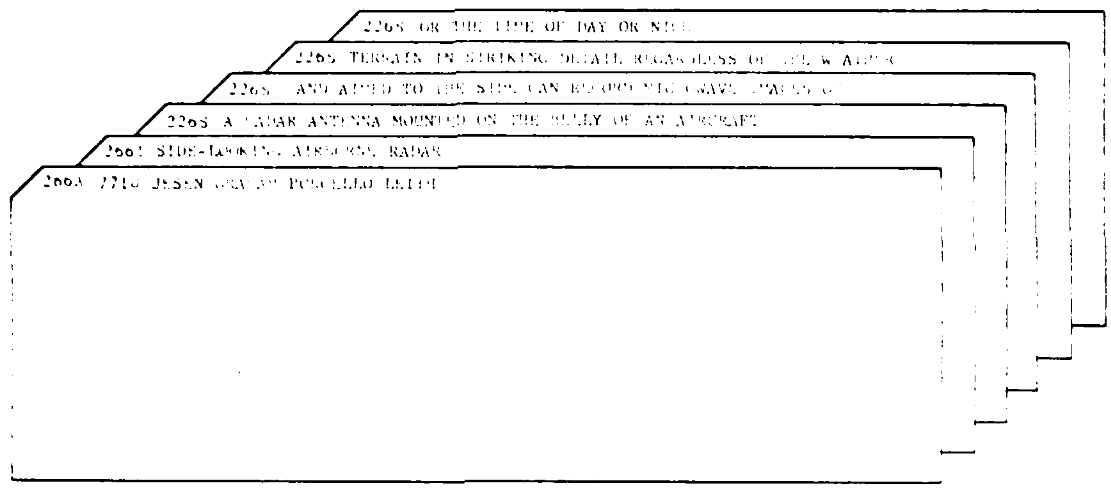
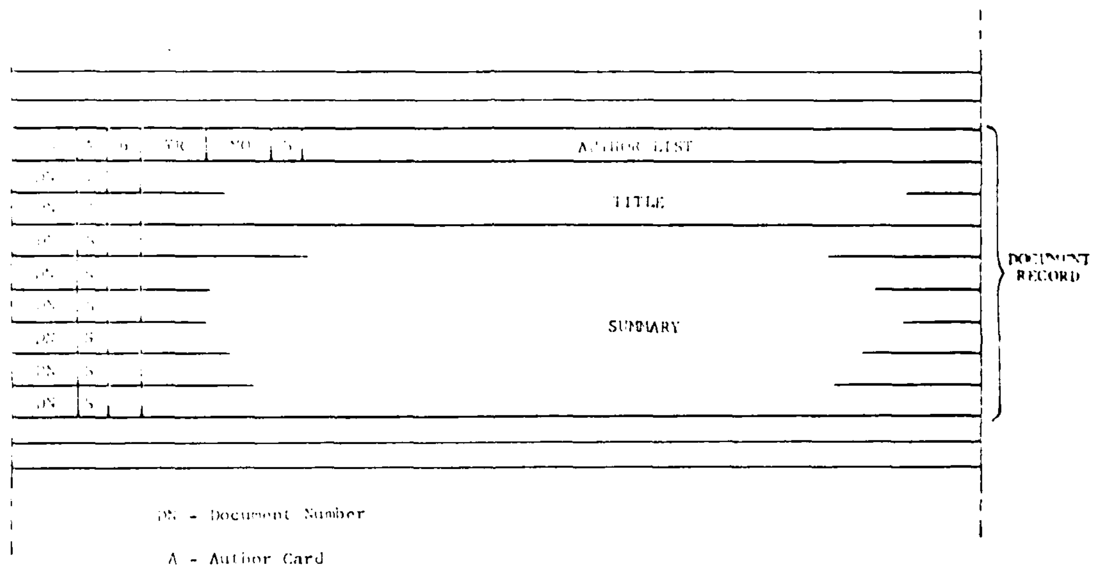


Figure 2-10. SA Data Base Document Format



DN - Document Number

A - Author Card

T - Title Card

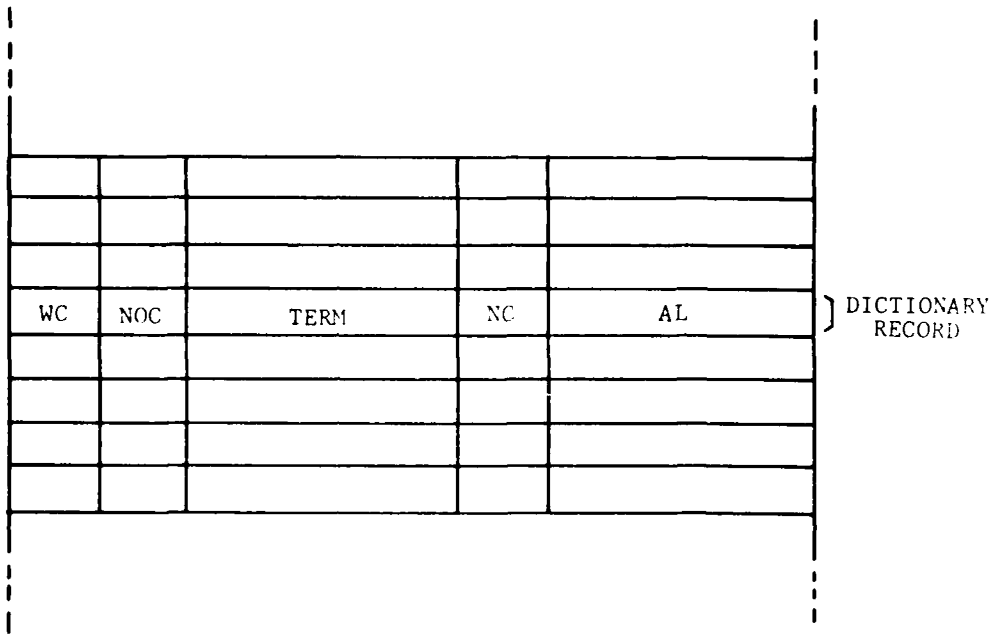
S - Summary Card

YR - Year of Publication

MO - Month of Publication

b - Blank

Figure 2-11. SA Document File Record Format



WC - Term Identification Number

NOC - Number of Occurrences

TERM - First 16 Characters of the Term

NC - Total Number of Characters in the Word

AL - If NOC = 1, AL = (DN,WN) where

DN - Document Number

WN - Word Number

If NOC 1,

AL - Address of the Term's Occurrence List

D5040

Figure 2-12. SA DICTIONARY File Record Format two meanings. If the term has a single occurrence, the AL field contains the document number, word number pair for the term. With multiple-occurrence terms, AL is the address in the OCCURRENCE file of the beginning of the occurrence list for the term.

The OCCURRENCE file is a list of occurrence lists as indicated by Figure 2-13. The lists are delimited by a special code, and the term identification number and number of occurrences are included in the record format.

2.5.3 Experimental Data Base Characteristics

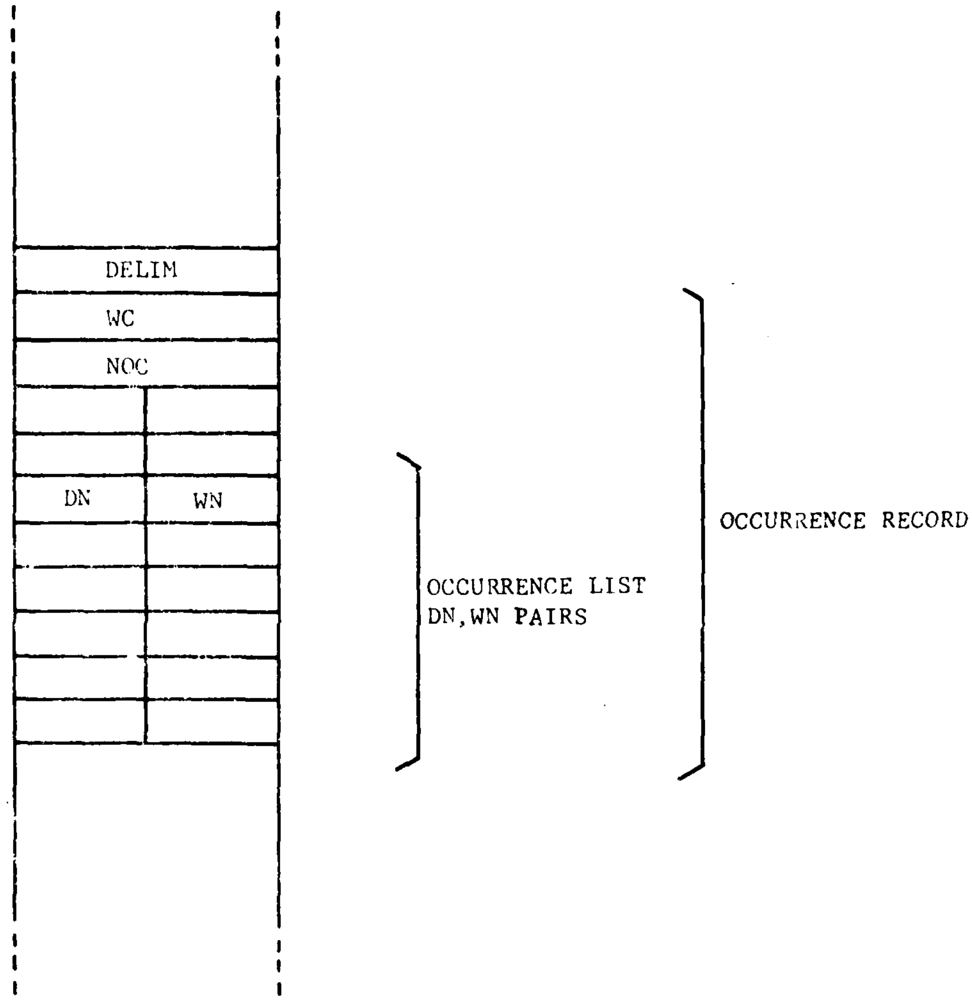
The 192 documents of the SA data base contain a total of 7153 words and produced a dictionary having 2551 dictionary terms. For comparison with the CIIO data base, the distribution of occurrence list lengths was measured, and the probability density function shown in Figure 2-14 and approximated by Equation 2-2 was derived.

$$\Pr\{n\} = 0.678 \times n^{-2.21} \tag{2-2}$$

Although the CIIO and SA data bases differ greatly in size (three orders of magnitude) their characteristics shown in Figures 2-7 and 2-14 agree very well. In the smaller data base a higher frequency of single-occurrence terms and lower probabilities of high-occurrence terms are to be expected.

2.5.4 Growth Characteristics

The SA data base provided a useful tool for studying the growth characteristics of document retrieval data bases. First, the growth in the number of dictionary terms as a function of the number of document words was observed as documents were added to the data base. This growth behavior is illustrated in Figure 2-15 and, as can be seen, is well behaved. An expression for the curve of Figure 2-15 is given by Equation 2-3, where W is the number of document words in the data base and T is the resulting number of dictionary terms.



DELIM - Delimiter

WC - Term Identification Number

NOC - Number of Occurrences

DN - Document Number

WN - Word Number

D5041

Figure 2-13. SA Occurrence File Record Format

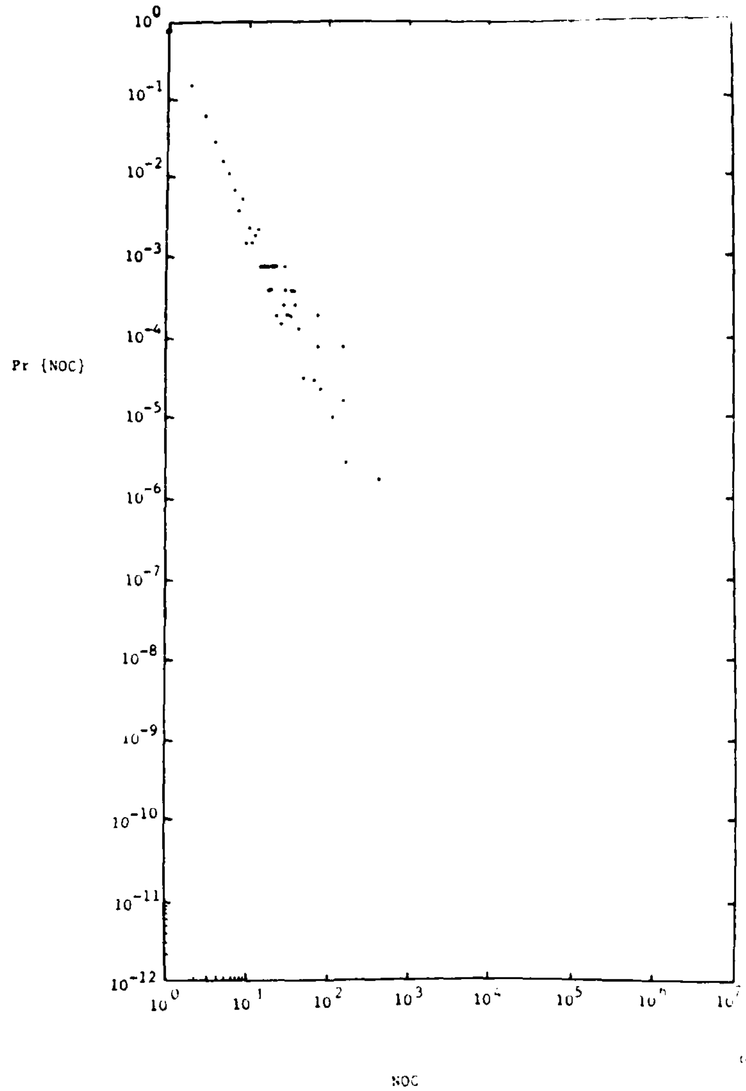


Figure 2-14. PDF of Number of Occurrences (SA Data Base)

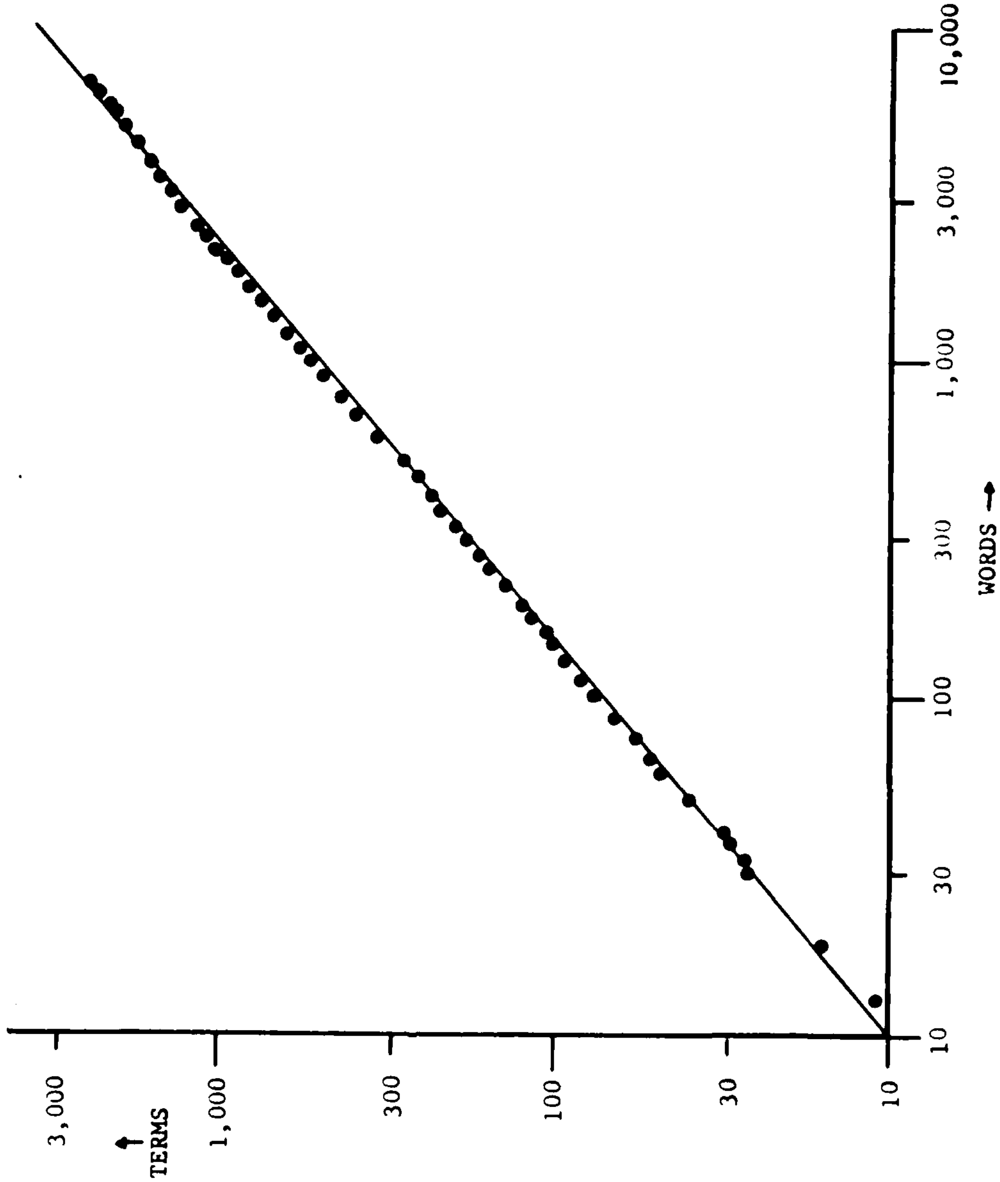


Figure 2-15. Dictionary Growth

D4143

$$T = 1.41\ W^{0.85}$$

The second method of investigating the data base growth characteristics was to build data bases with varying numbers of documents and observe their characteristics. The data bases studied are indicated in Table 2-4. From this experiment, the growth characteristics in the numbers of infrequent terms, shown in Figure 2-16, were observed. The dictionary growth is also shown in Figure 2-16, for comparison. Note in particular the decreasing probability for single-occurrence terms as the data base grows. Expressions for the curves shown in Figure 2-16 are given by Equations (2-4) through (2-7), where T_i is the number of dictionary terms having i occurrences.

$$T_1 = 1.8\ W^{0.78} \tag{2-4}$$

$$T_2 = 0.19\ W^{0.87} \tag{2-5}$$

$$T_3 = 0.035\ W^{0.95} \tag{2-6}$$

$$T_4 = 0.011\ W^{0.99} \tag{2-7}$$

The growth in occurrence list lengths for the more frequently occurring terms is shown in Figure 2-17. The growths in occurrences are shown for the six most frequent terms, together with a sampling for some less frequent terms. An expression for the broken curve of Figure 2-17 is given by Equation 2-8 where N is the number of occurrences.

$$N = 0.120\ W^{0.94} \tag{2-8}$$

All the data base growth characteristics investigated and reported here have shown well developed characteristics, indicating that data base growth is highly predictable, more so than might have been anticipated. This information will be very useful in developing details of a data base organization conducive to growth. For example, it is evident that the more frequent terms show the most rapid growth, and spare memory should be available

TABLE 2-4. SA DATA BASE SIZES

NUMBER OF DOCUMENTS	NUMBER OF WORDS	NUMBER OF TERMS
7	270	170
12	462	264
16	612	344
24	884	495
32	1180	633
48	1719	816
60	2236	1075
80	2773	1254
100	3772	1560
120	4518	1801
150	5647	2129
192	7153	2551

C8-0656

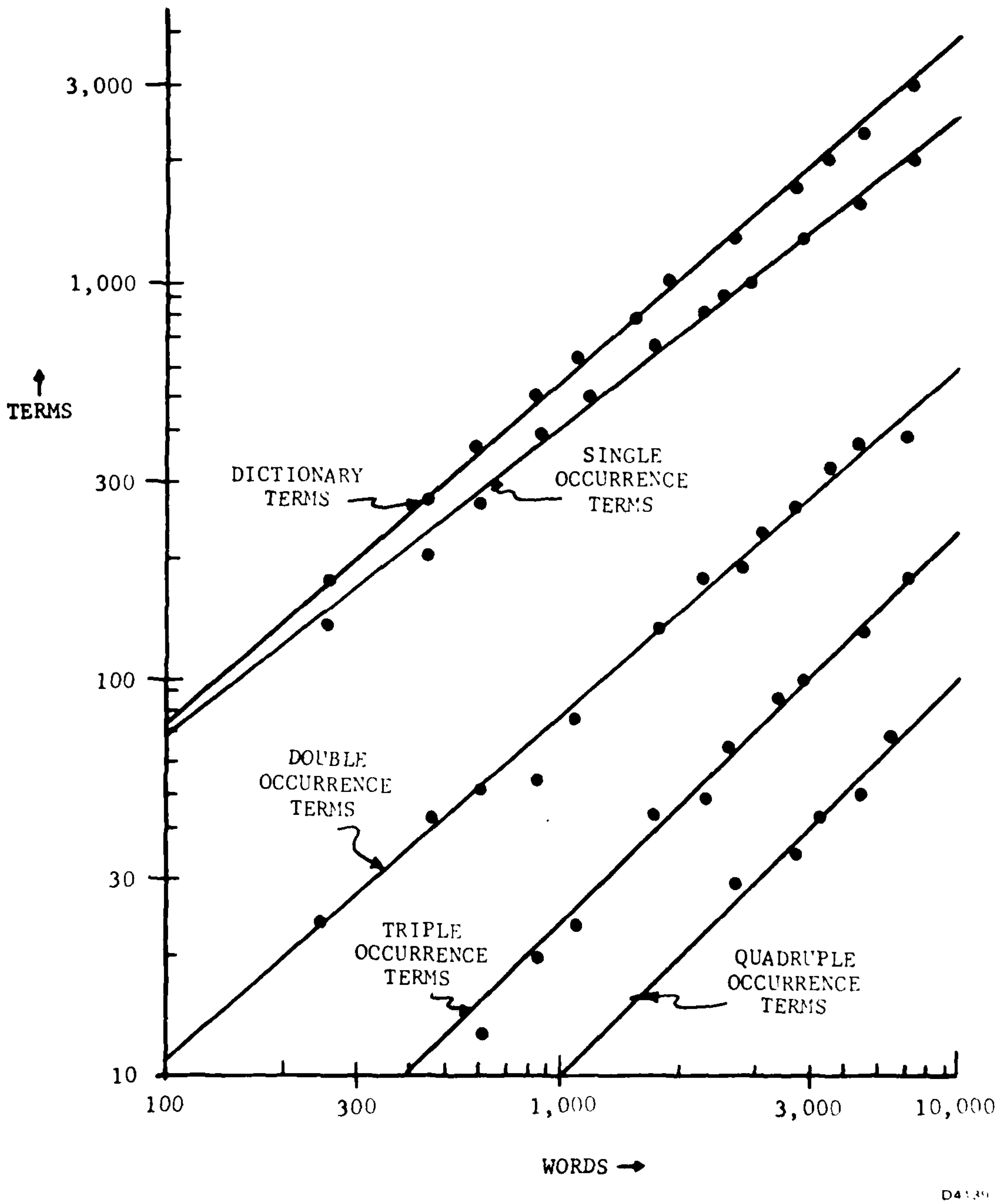


Figure 2-16. Dictionary Growth for Infrequent Terms

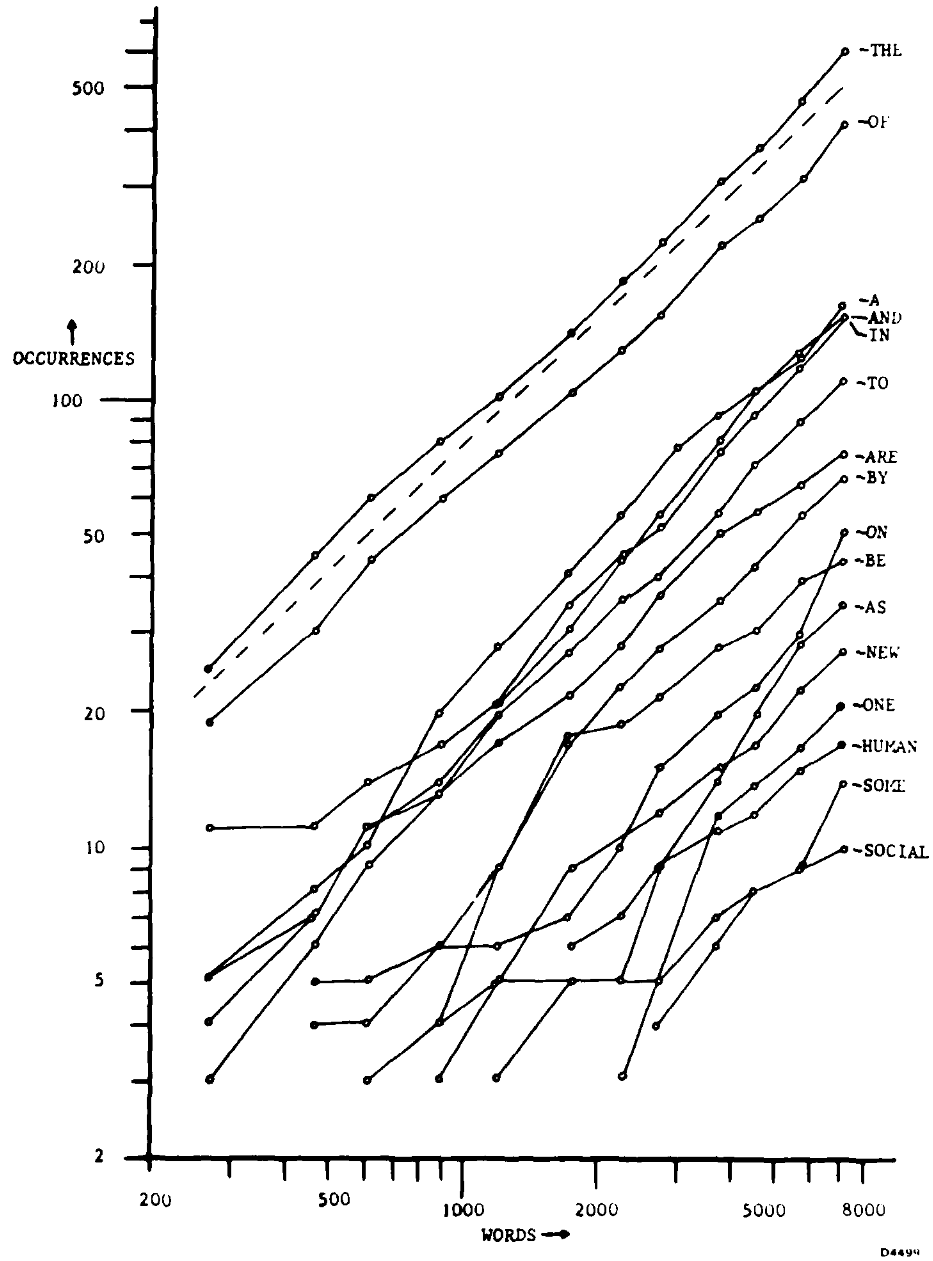


Figure 2-17. Dictionary Growth Characteristics for Frequent Terms

at the end of each such occurrence list to reduce the frequency of memory reallocation. While the infrequent terms do not grow rapidly on an individual basis, they do as a class. Thus, the memory allocation strategy should allow space for new, short occurrence lists.

3.0 RECONFIGURABLE ARRAYS OF FLEXIBLE PROCESSORS

In this section, the major hardware components from which reconfigurable arrays are synthesized are described. The Control Data reconfigurable arrays consist of microprogrammable processors interconnected by one or more Ring Communication Systems, and connected to auxiliary memory systems. These arrays are reconfigurable because their communication systems are programmable. Two types of processors can be used in the arrays, the existing Flexible Processor, and the Advanced Flexible Processor. Both machines are described in this section. The memories which can be attached to the processors include both high-speed random-access memory banks, and disk memories. An array is normally controlled and interfaced with its environment through a conventional computer, such as a CDC 1700 computer

The Laboratory Reconfigurable Array used as a demonstration system for this study is described in Section 4.0 of this report.

3.1 THE FLEXIBLE PROCESSOR

The Control Data Flexible Processor is a digital microprogrammable processor which has been designed specifically for multidimensional array processing. Since the units are microprogrammable, they combine the advantage of special-purpose hardware in efficiency with the flexibility for software changes.

Each Flexible Processor can be used as a building block in a modular system design where each Flexible Processor is accordingly programmed with appropriate algorithms to perform specific tasks which are assigned to it. Moreover, programming of the various Flexible Processors in an array can be changed dynamically under control of a processing system which calls prescribed programs from memory, as from a rotating disk file. The size of the Flexible Processor has been effectively chosen to permit efficient construction of a complete system tailored to meet project requirements.

3.1.1 Hardware and Software Description

The Flexible Processor is a special-purpose computing unit featuring high arithmetic computation rates, considerable character-handling capabilities, an advanced input-output structure and semiconductor register file memories for data storage. Each unit features arithmetic logic capable of a 125-nanosecond addition of 16-bit or 32-bit operands, a 250-nanosecond fixed-point multiply of 8-bit bytes, and a 1.125-microsecond fixed-point multiply of 16-bit operands (Table 3-1).

The Flexible Processor is a dual bus organized machine with a modular set of functional units centered around a control section, including conditional instruction execution networks, interrupt hardware, and featuring a read/write semiconductor micromemory (Figure 3-1). The micromemory word contains both direct control functions and decoded bus controls. Movement of operands among the functional units is particularly effective in applications which require a high data throughput.

3.1.2 Functional Units

Four basic functions comprise the operation of the Flexible Processor: Arithmetic Logic Unit, Hardware Array Multiplier, Register File Storage, and Input/Output. They are described in the following paragraphs.

Arithmetic Logic Unit

This unit is capable of 39 unique arithmetic and logic operations. Shifting of all input feeder registers, right or left, with a selection of inputs to the registers during the shift instruction, and a "scale" capability on the output of the ALU extend the capabilities of this functional unit. The ALU may be configured in a 16-bit or 32-bit format (one and two cards, respectively).

TABLE 3-1. CDC FLEXIBLE PROCESSOR CHARACTERISTICS

- MICROPROGRAMMABLE - RANDOM ACCESS MICROCONTROL MEMORY
- 32-BIT OR 16-BIT WORD LENGTHS
- ARRAY HARDWARE MULTIPLIER
- 16 LEVEL HARDWARE PRIORITY INTERRUPT MECHANISM - 3 LEVEL MASK CAPABILITY
- SPECIALIZED LOGIC FOR SQUARE ROOT AND DIVIDE
- 8 mHZ FILE BUFFERED WORD TRANSFER RATE - 16 WORD BY 32-BIT OR 16-BIT INPUT FILE BUFFER
- 2 mHZ DIRECT MEMORY ACCESS WORD TRANSFER RATE
- 1 mHZ REGISTER-BUFFERED WORD TRANSFER RATES
- DUAL 16-BIT INTERNAL DATA BUS SYSTEM
- 0.125 us CLOCK CYCLE
- 0.125 μs 32-BIT ADDITION: 0.250 μs BYTE MULTIPLICATION
- REGISTER FILE CAPACITY UP TO 4128 16-BIT WORDS
- HARDWARE NETWORK FOR CONDITIONAL MICROINSTRUCTION EXECUTION - 4 MASK REGISTERS AND A CONDITION HOLD REGISTER

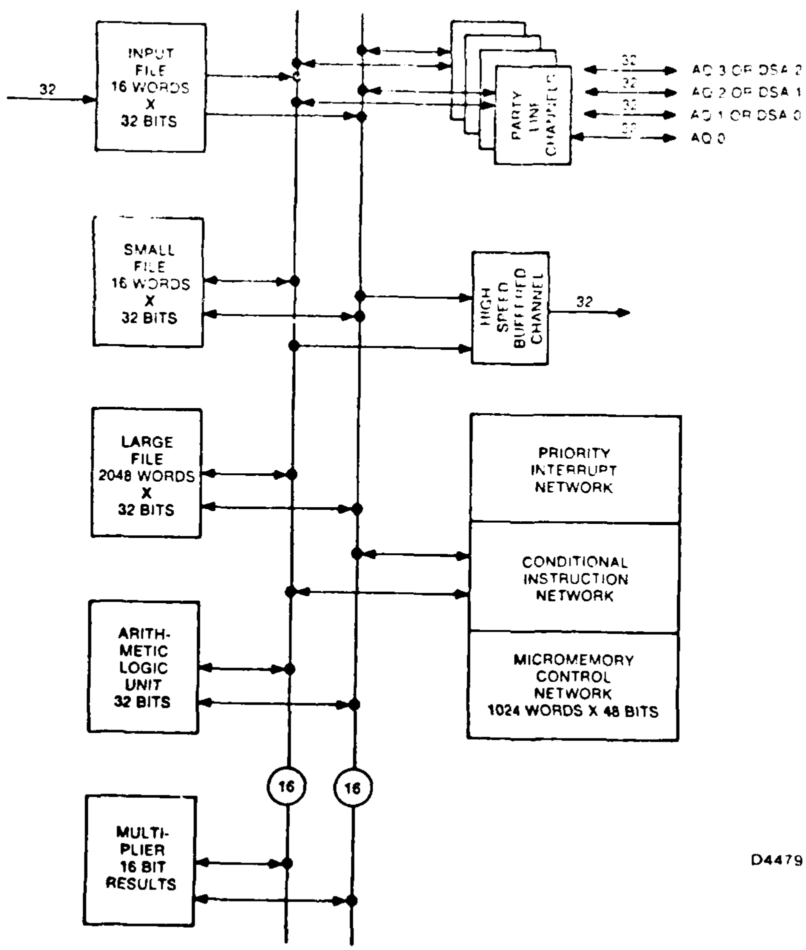


Figure 3-1. Data Path Organization in the CDC Flexible Processor

Hardware Array Multiplier

The Hardware Array Multiplier is a modular arithmetic function which is capable of eight by eight multiples, with each 8-bit operand directly selectable from two 16-bit input registers. Multiplication is performed asynchronously to the control unit, allowing other instructions to be executed while the multiplication is taking place.

Register File Storage

The internal Register File Storage is semiconductor memory providing both Temporary and Large File functions. The Temporary File provides 16 words of 16 bits each for each bus. This file has separate read and write address registers, allowing simultaneous read and write capability. The Large File provides a maximum storage of 2048 words of 16 bits each for each of the two buses.

Input/Output

The Flexible Processor has three types of external data transmission functional units available. These are:

- Bi-Directional Party-Line Channel (AQ)
  This channel is compatible with CDC 1700 peripheral equipment and has a maximum data rate capability of one megaword per second (16-bit data word). There is a maximum capacity of up to four of these channels; however, they share space with the DSA channel.

- External Mass Memory Access Channel (DSA)
  This channel is primarily intended for use with an MOS semiconductor mass memory. This channel has a data rate capability of two megawords per second (16-bit data word). The maximum addressable memory contains over one million 16-bit words (20 address bits, four of which are used for bank selection). There is a maximum of three of these channels; however, one channel is forfeited for

each additional AQ channel in the processor.

- Inter-Flexible Processor Communication Channel

This functional unit permits asynchronous data transmission between Flexible Processors at eight megawords per second with a maximum word length of 32 bits. In terms of the bit transfer rate, the maximum communication rate between Flexible Processors is 256 million bits per second. This communication channel is comprised of two dependent sections. One section is located in the sending Flexible Processor, and the other section is in the receiving Flexible Processor. The receiving section has a 16-word memory capable of simultaneous read and write operations enabling transmission asynchronous to the receiving Flexible Processor's usage of the operands. Each Flexible Processor has a maximum capacity of two 16-bit word sending sections and two 16-bit word receiving sections. In addition, each sending section can communicate with as many as four receiving sections. This channel can also be replaced by ports for a Ring Communication System which will be described later.

3.1.3 Miscellaneous Hardware Capabilities

Miscellaneous hardware capabilities of the Flexible Processor are Loop Counters, Return Jump File, Micromemory Control Storage, and optional external memory. These capabilities are described in the following paragraphs.

Loop Counters

Four counters are provided for software problems involving iterative computation. There are three 8-bit counters and one 16-bit counter with two compare registers associated with each counter.

Return Jump File

A return jump file of 16 words is used for subroutine exits, interrupt return, and iterative loop returns. This is a push-down, push-up file capable of returning control to a main program through a maximum of 16 nested loops or subroutines.

Micromemory Control Storage

The micromemory consists of 48-bit words of random access read and write semiconductor memory with increments from 256 words to 1024 words for program storage.

External Memory

A MOS semiconductor memory bank can be added to a basic Flexible Processor by adding a DSA channel in the Flexible Processor and one or more racks of memory. This external memory is expandable in 8192-word increments from 8192 16-bit words to 65536 16-bit words per bank with a maximum of 16 banks per DSA channel. This memory has the capability of multiport operations with any one port having a data rate capability of two megawords per second, and multiport operation of three megawords per second. Faster memory capabilities for any Flexible Processor can be achieved by three DSA channels in the Flexible Processor communicating with separate ports of unique memory banks.

3.1.4 Software

As stated previously, the Flexible Processor is a microprogrammable processor with read and write random access instruction memory. Thus, through this memory the Flexible Processor is encoded to compute a desired algorithm, and the memory can be rewritten subsequently to perform different algorithms if it is so desired.

The instruction is a 48-bit word containing both decoded and direct control fields. The instruction word defines the following operations:

- Simultaneous control of the Arithmetic Logic Unit, Multiplier, Jump Stack, Loop Counter, Conditional Execution, and the two buses.

- Independent shifting of three registers

- Simultaneous control of four Input/Output Channels

A cross-assembler, MPASS2, has been written to aid in the generation of microprograms for the Flexible Processor. This assembler runs on the CDC 1700 computer system utilizing a data deck that has a one-to-one correspondence between the micromnemonics and the microinstruction control fields. A symbolic addressing capability of the assembler also eases the writing of microprograms.

Other available software routines are:

- Library routines for generation of a microprogram library on disk file.

- Loader routine for loading microprograms into the Flexible Processor from the library disk file.

- Diagnostics for all functional units in the Flexible Processor.

- Display Station Monitor for hardware and software checkout of the Flexible Processor and microprograms.

3.2 THE RING COMMUNICATION SYSTEM

A reconfigurable array of Flexible Processors is constructed using a Ring Communication System developed by Control Data under a RADC contract (See Final Technical Report, RADC-TR-78-57, March, 1978). The ring communication system for Flexible Processors is discussed briefly in this section.

The ring hardware (See Figure 3-2) is capable of transferring data between any of the processors in an array. The central ring consists of paths for

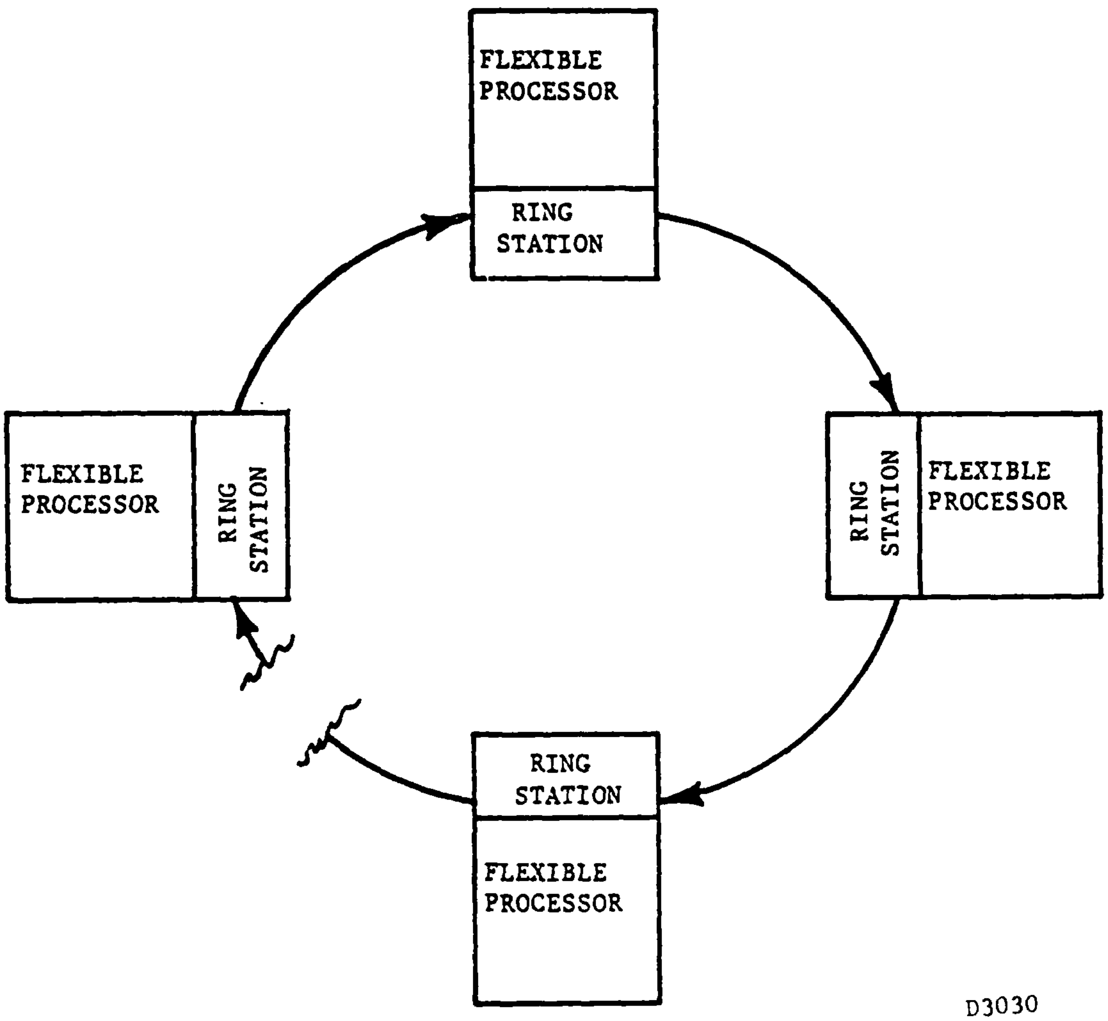


Figure 3-2. The Basic Ring Concept

data, destination address and flag bits. The ring will interface with array processors through ring stations, which contain provisions for entering data packets on the ring, removing and buffering incoming packets intended for that station, passing packets with different destinations through the station, and degradation ability in the event of hardware failure. A means is provided to uniquely identify each ring station and to alter that identifier under system software control. Each station has the ability to buffer incoming packets intended for that station, and to provide a large amount of the control and addressing functions necessary to place packets on the ring, relieving the associated processor of these "housekeeping" chores. Transfer between the ring stations is synchronous in order to achieve as high a data rate as possible.The ring has the ability to transfer a packet every 125 nanoseconds (a packet consists of 32 bits of data and 10 bits of address, function and flag control). The ring station is configured in a manner which will allow replacement of the present FP-to-FP high-speed channel found in the standard Flexible Processor. Each Flexible Processor will be able to contain two ring stations.

The ring station (See Figure 3-3) is divided into several major sections: the input section, the output section, and the control section. Data flow through these sections is summarized as follows: A data packet, consisting of a "word" of information transmitted in parallel from the previous ring station, arrives at the input of the ring interface every ring period, where it is latched in a register. A given input packet will be one of four types. It will contain information intended for receiving station, information intended for some other station on the ring, information intended for the receiving station and also intended for some other station on the ring, or no information, which would represent an empty slot on the ring.

The packet type is determined by the control section from data contained in the packet. Packets intended for the receiving station are routed by the control section to the input section where they are stored in an input file, which can be sourced by the Flexible Processor bus. Packets intended for other stations are routed unchanged through the interface to an output register, where they are latched for transmission to the succeeding station

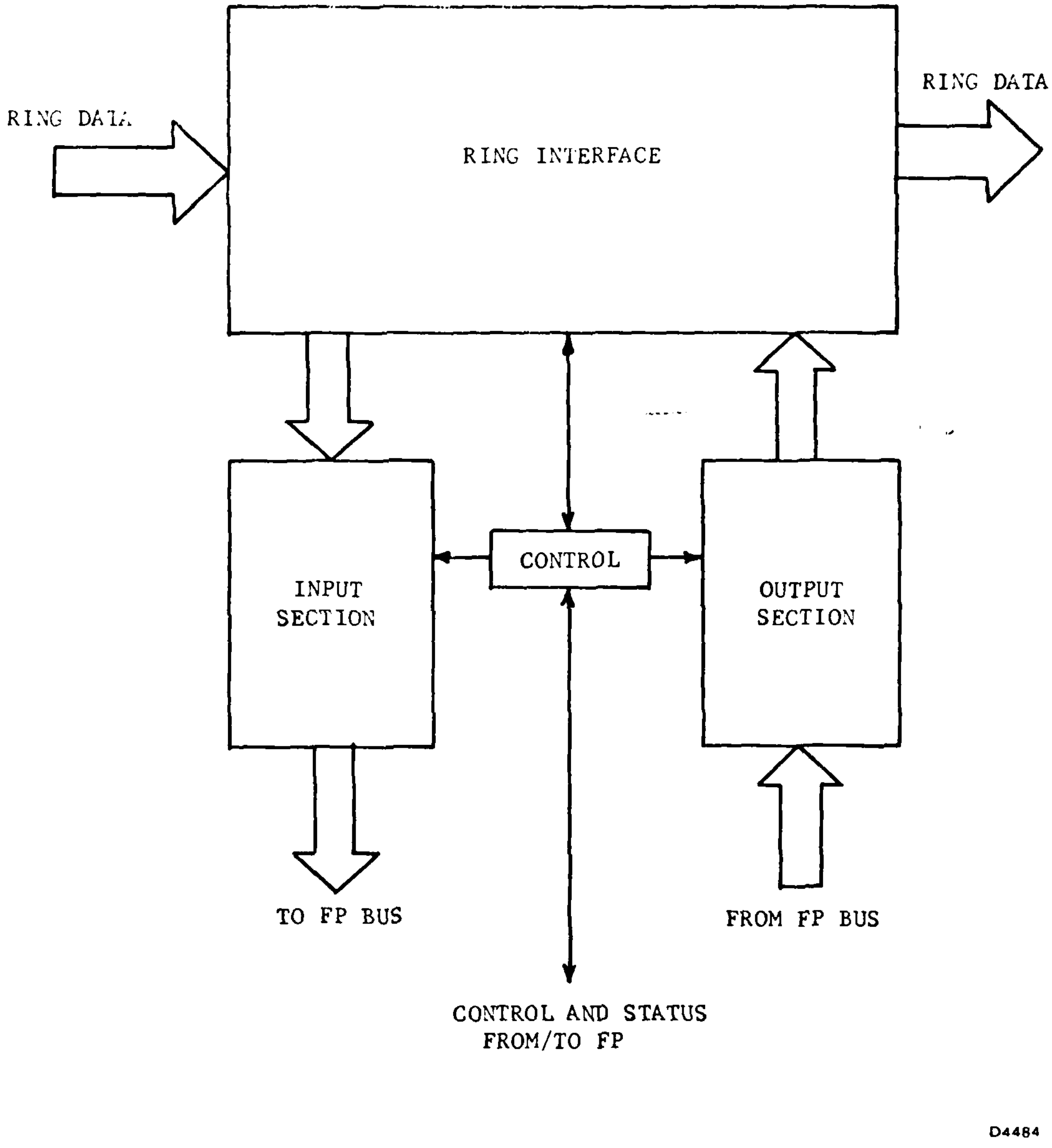


Figure 3-3. Ring Station

during the subsequent ring period.

An empty packet entering the interface, or a packet which is removed from the ring and placed in the input file, creates an opportunity for the station to place a new packet on the ring. If the processor has placed information in the output buffer and it is ready to be transmitted at this time, the data will be routed from the output buffer, along with destination and control information, to the output register of the ring station where it is transmitted to the next station during the next cycle.

Control signals from the Flexible Processor notify the ring station of information to be placed in the output buffer for ring transmission and allow access by the processor to data in the input file. The station returns signals to the processor regarding the status of input blocks and the availability of empty slots for output.

3.3 THE DATA CHANNEL CONTROLLER

The Control Data Data Channel Controller (DCC) provides a mechanism for connecting the Flexible Processor of a Reconfigurable Array with high capacity, secondary memory devices. Each Data Channel Controller supports record transfers of data between a maximum of 16 Flexible Processors and 48 disk drives, representing up to 10^{11} bits of storage. More than one Data Channel Controller can be attached to a Reconfigurable Array for increased storage capacity or transfer rate.

The organization of a Data Channel Controller is illustrated in Figure 3-4. The DCC is built around a random access buffer memory of 128K bytes which is used to buffer records between the Flexible Processors and the disks. The control element (CE) of the DCC is connected to the Reconfigurable Array in two ways. A set of ports which are compatible with Flexible Processor DSA channels is provided for the transfer of data. In addition, a port compatible with an AQ channel provides a path for the transfer of commands and other control information.

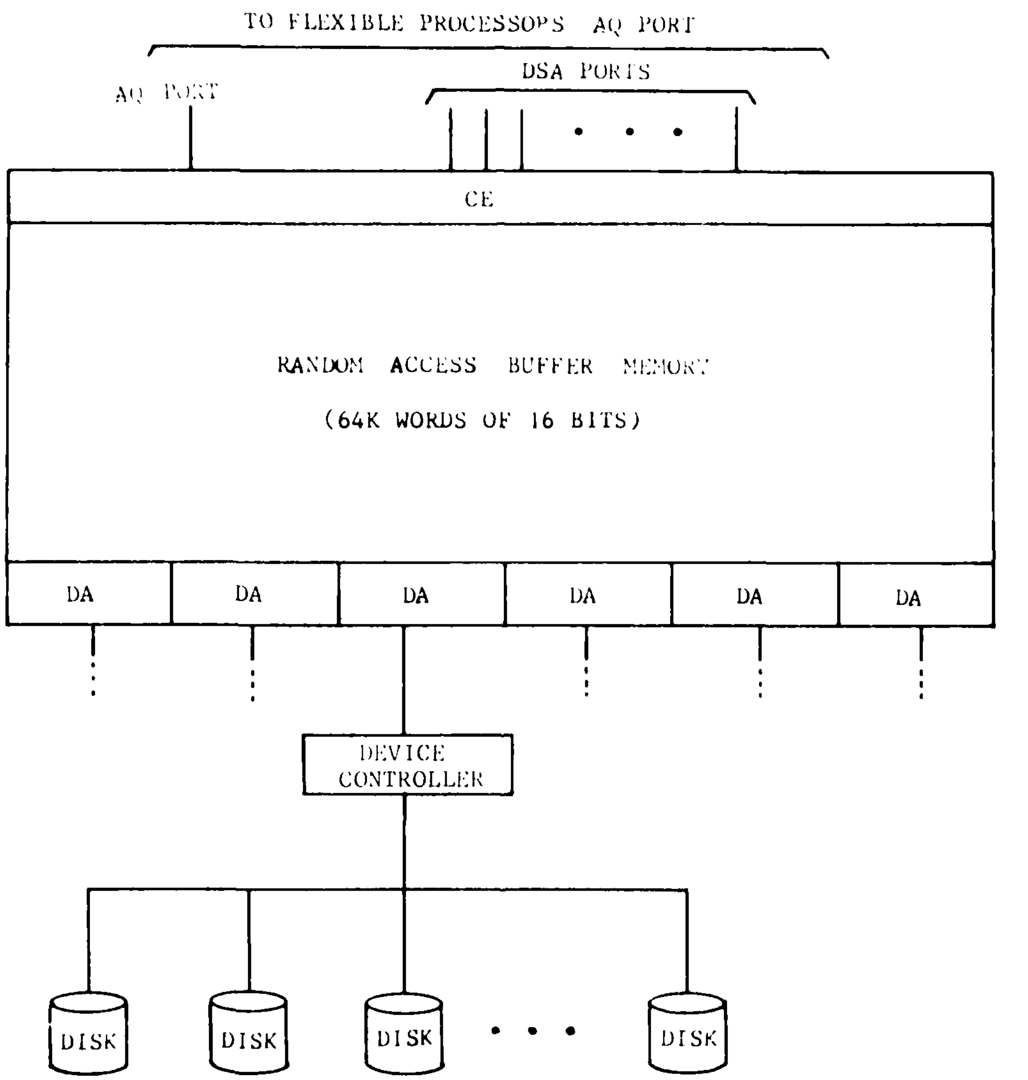


Figure 3-4. Data Channel Controller Organization

On the other side of the buffer memory, the DCC can have up to six disk adapters (DA) which govern transfers of records between the buffer memory and the disk controllers.

3.4 THE ADVANCED FLEXIBLE PROCESSOR

This section presents a general description of the Advanced Flexible Processor (AFP). The AFP consists of a number of functional units which communicate with each other via a crossbar network. The processor configuration, as shown in Figure 3-5, has typical mix of functional elements.

3.4.1 The Crossbar-Data Switch

In Figure 3-5, several lines are shown as entering and leaving an element called the crossbar. Each of these lines is a data path 16 bits in width. The crossbar provides the capability to transfer data on many paths simultaneously. In fact, data on 16 16-bit paths can be transferred into the corssbar and data transferred out of 18 data paths on one instruction cycle. This provides a total communication bandwidth of about 12.8×10^9 bits per second. The crossbar network thus provides a very high band width network for internal communications.

The generalized nature of the crossbar data path has many benefits. The programmer has complete control over data routing between all functional elements on every machine cycle. This gives the programmer the capability to control, in detail, the data flow and machine execution.

3.4.2 The Processor Characteristics

The various general characteristics of the Advanced Flexible Processor are listed in Table 3-2 below.

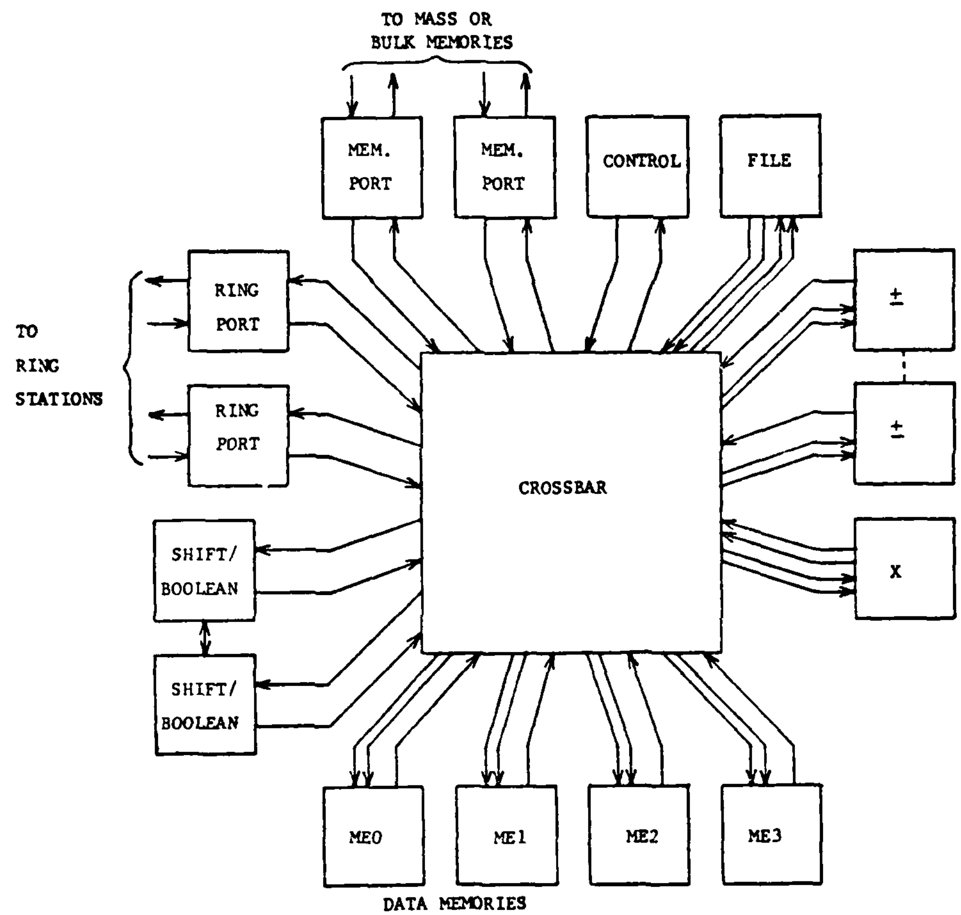


Figure 3-5. Basic Advanced Flexible Processor

TABLE 3-2. AFP FEATURES

- 50 MIPS - 20 NS Per Microinstruction
- LSI ECL Technology
- Multiple Functional Units
- Generalized Internal Data Path - Crossbar
  - 256 Bits per Cycle
- Internal Configuration Flexibility
- Add-On Capability
- Microprogrammed
- Multiway Branching
- Ring System - Interprocessor Communication

One of the most important features of the AFP design is the ability to vary the structure of the processor without affecting the machine instruction set. This provides great hardware flexibility. Specialized units needed for a particular application to obtain higher performance can easily be added. One such specialized unit is the Associative Memory Unit described in Section 3.5.

3.4.3 Functional Unit Capabilities

The primary capabilities of the hardware functional units are listed in Table 3-3.

These capabilities are explained in greater detail in the processor specification. It should be noted that, in general, each functional unit has considerable processing capability and that multiple operations are performed by most of the units. For example, the Shift/Boolean unit can perform both the shift operation and the Boolean operation in one machine cycle. The multiply unit has the capability to perform either a 16 x 16 multiply, or two simultaneous 8 x 8 multiplies. The file unit is a particularly high performance unit having the capability to perform two read and two

TABLE 3-3. FUNCTIONAL UNIT CAPABILITIES

INTEGER ADD	ADD/SUBTRACT, 16 OR DUAL 8-BIT, 1's OR 2's COMP, 32-BIT NETWORK
MULTIPLY	16 x 16, DUAL 8 x 8, INTEGER BYTE PRODUCTS, 2's COMP
SHIFT/BOOLEAN	32-BIT RIGHT OR CIRCULAR SHIFT, 16 BOOLEAN OPERATIONS
FILE	2 SETS OF 8-WORD x 16-BIT REGISTERS, 2 16-BIT READS AND 2 16-BIT WRITES PER CYCLE
DATA MEMORY	1024 16-BIT WORDS, 16 16-BIT REGISTERS DIRECT AND INDIRECT INDEX ADDRESSING, AUTOMATIC INDEX INCREMENT/DECREMENT
CROSSBAR PORT	16-BIT OUTPUT AND 16-BIT INPUT DATA CONNECTIONS TO THE CROSSBAR
RING PORT	16-BIT DATA I/O WITH RING AND PROCESSOR, 16-WORD INPUT AND 16-WORD OUTPUT BUFFERS FORCED TRANSFER TO ALL PROCESSOR MEMORIES
XMAU	EXTERNAL MEMORY ACCESS 128-BIT MEMORY I/O AND 16-BIT PROCESSOR I/O, MAX ADDRESS 3 BYTES

write operations on the same machine cycle.

3.4.4 Microinstruction Format

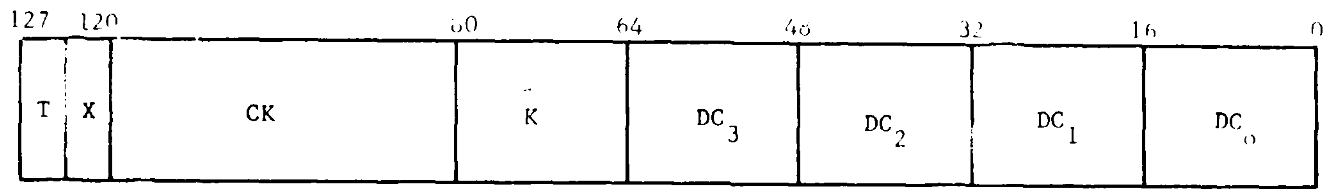
The AFP is microprogrammed processor controlled by two types of instructions as shown in Figure 3-6. The first type of instruction, Type 0, is the most widely used instruction. The second type of instruction is used for loading a memory unit which provides detailed control of crossbar data path routing. This type of instruction is used primarily when loading in a new application program.

In the Type 0 instruction, each of the four DC fields controls the functional units. Each of the fields can control any of the functional units. This provides a very generalized instruction format. Also, the individual functional units can be changed without affecting the instruction format. The constant field (K) provides 16 bits of data which can be routed to any of the functional units directly from the microinstruction.

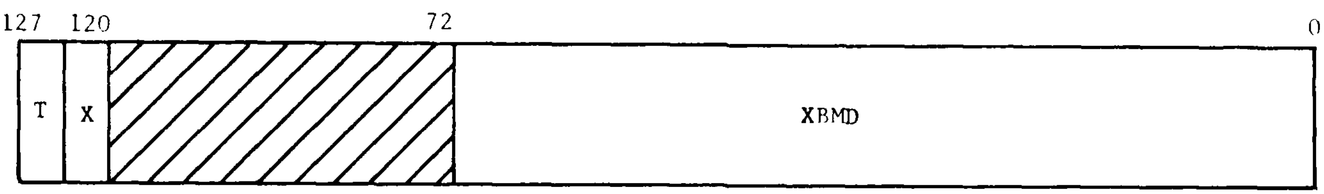
The clock field (CK) controls the input feeder registers in each of the functional units. The clock field provides a unique and powerful capability to set-up multiple pipelines within the processor such that all 16 units can be used simultaneously.

The X field addresses the memory unit which supplies control information to the crossbar. The T field selects one of two types of instruction format. More detail on the instruction format is provided in Figure 3-7.

The clock field controls feeder registers in the functional units. In general, two bits in this field are assigned to each functional unit. A bit "1" allows the input registers to be clocked and loaded with new data. Each time the data input to the functional unit changes, even though the control information is the same, the unit will generate a new type of data result. Control of the function to be executed by the unit is changed by use of a DC field.



Type 0 Instruction



Type 1 Instruction

D4485

Figure 3-6. AFP Instruction Formats

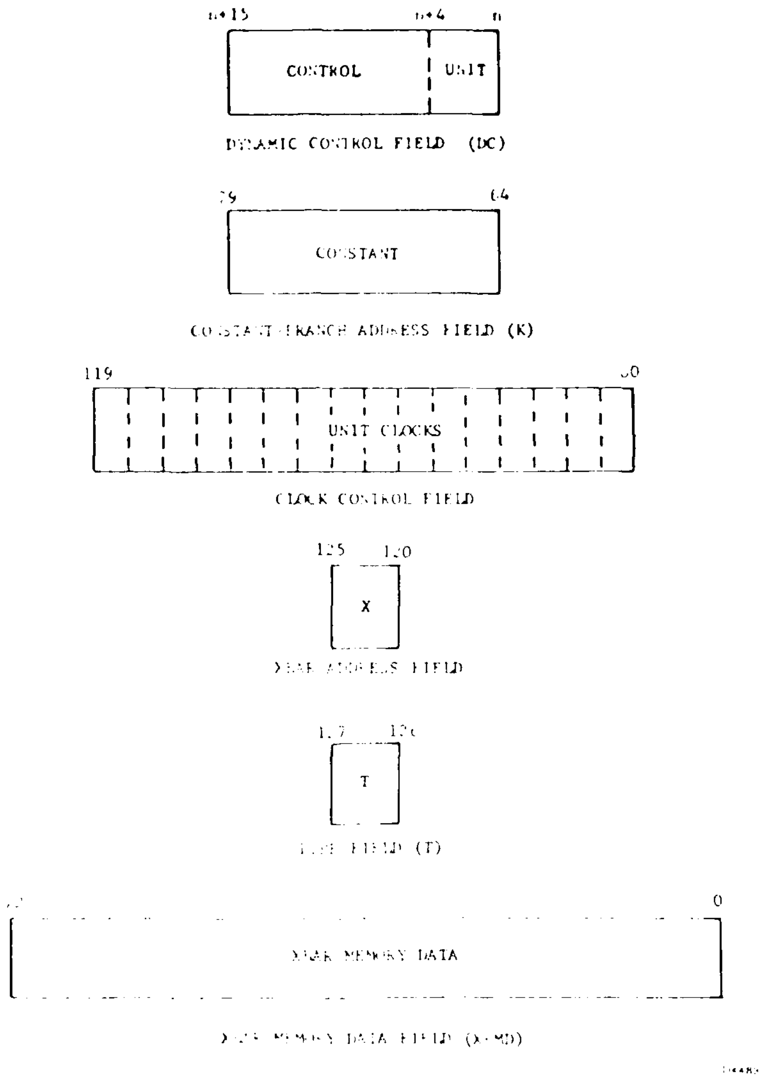


Figure 3-7. AFP Instruction Subfields

3.5 THE ASSOCIATIVE ADD-ON UNIT

The Associative Add-On Unit design consists of 256 associative processing cells and a microprgrammable controller organized as illustrated in Figure 3-8. The Exploratory Development Model was, however, populated with only thirty-two cells, as an evaluation tool during the study.

The Associative Unit (AU) is designed to operate as an add-on unit for both the Flexible Processor and the Advanced Flexible Processor as illustrated in Figure 3-9. The interconnections required between an Associative Unit and its host processor are summarized in Table 3-4. The internal differences between the Flexible Processor (FP) and the Advanced Flexible Processor (AFP) are compensated for by different interface units between an AU and its host. The AU requires a 16-bit data input path, an 8-bit control input path, a 16-bit data output path, and a number of other control signal lines. The AU signals can be changed once in every AU machine cycle. The functions of the signals defined in Table 3-4 are described further in a later section.

The internal machine cycle of the Exploratory Development Model Associative Unit has a period of 125 nsec. and is synchronized to the clock of the host processor.

3.5.1 The Associative Processing Cells

The Associative Processing Cells are partitioned into blocks of eight. A physical unit (card) consists of four cells together with their associated response logic (See Section 3.5.2).

The basic components of a cell are shown in Figure 3-10, and include a 256 x 4 Randon Access Memory (RAM), a 4-bit Arithmetic Logic Unit (ALU), and a cell control element. Each cell possesses a number of registers; the names and sizes of each register are listed in Table 3-5. All cells of an AU are common to a set of five buses which carry data and control information to

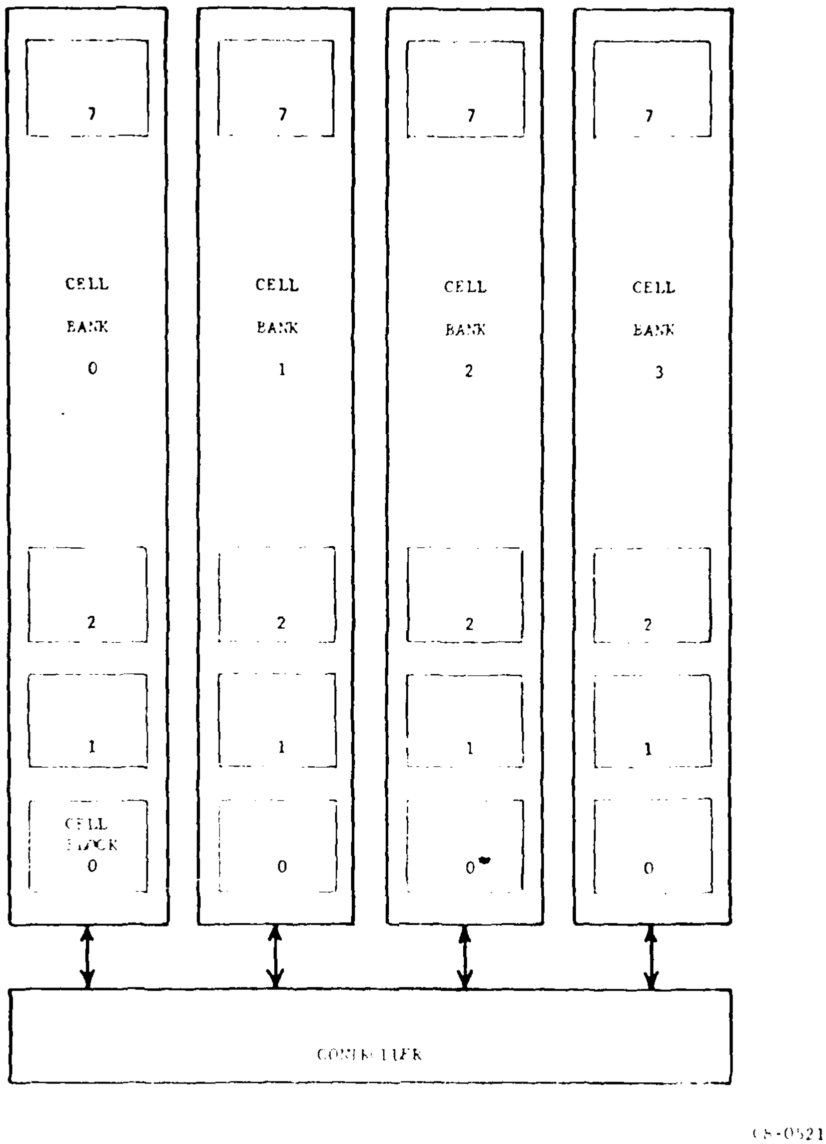


Figure 3-8. Organization of the Associative Unit

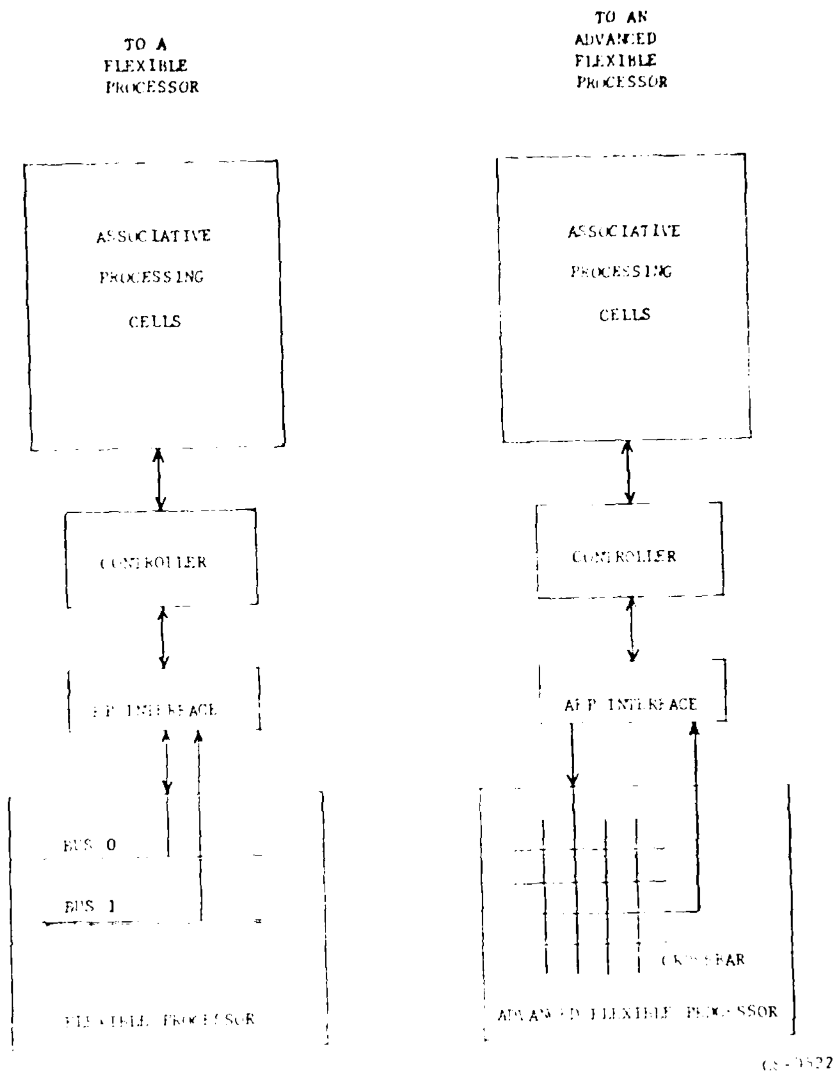


Figure 3-9. Associative Add-On Unit Interfacing

TABLE 3-4. AU/HOST INTERCONNECTIONS

FUNCTION		HOST			
		FP		AFP	
SYMBOL	DESCRIPTION	LINES	SOURCE/DESTINATION	LINES	SOURCE/DESTINATION
DI	Data Input	16	BUS 0	16	Crossbar
CC	Control Vector	12	BUS 1	12	Controller
CL	Clock Signals	3	Micromemory	3	Micromemory
DO	Data Output	16	BUS 0	16	Crossbar
ST	Status Vector	1	Condition	8	Conditions
IT	Interrupt	1	Interrupt	1	Interrupt
PH	Master Clock	1	Clock	1	Clock

C8-0523

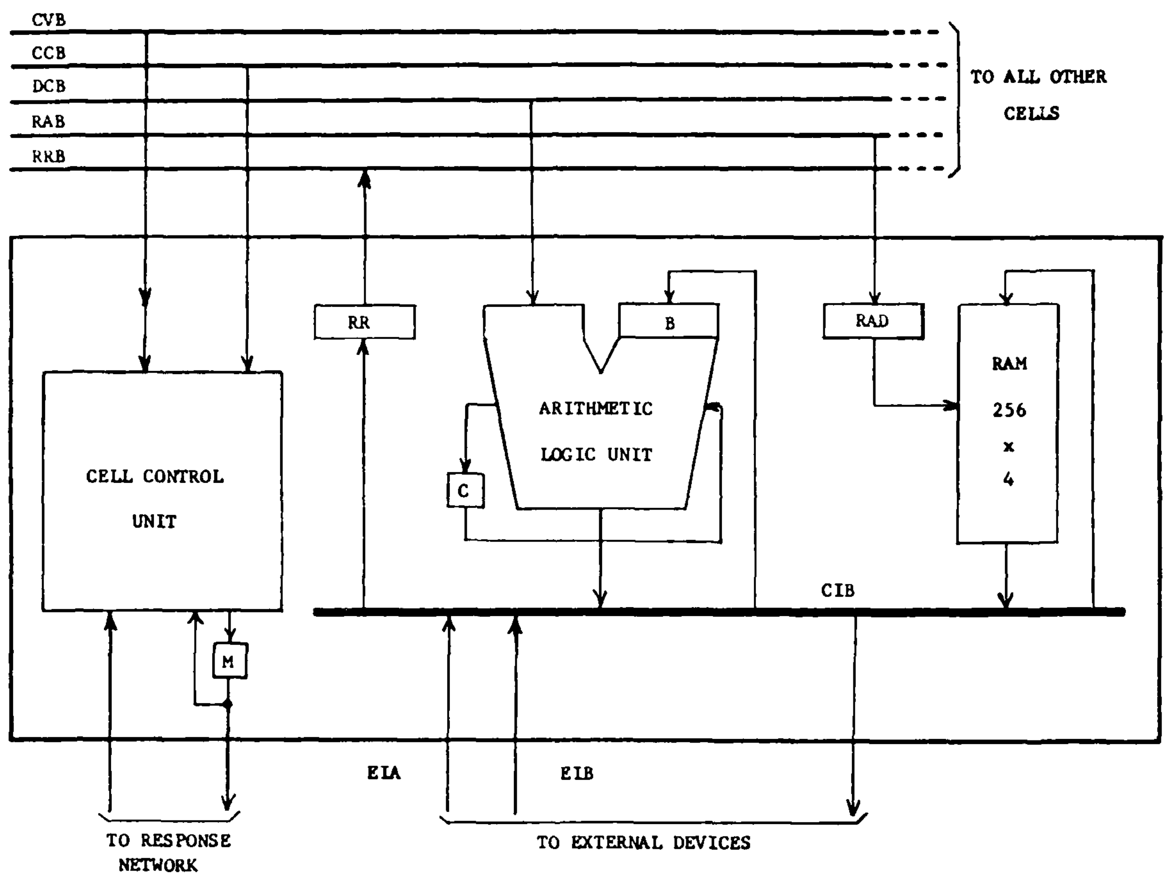


Figure 3-10. Associative Unit Cell Structure

TABLE 3-5. CELL REGISTERS

SIZE	NAME	USE
4	RR	Read Response Register - holds data for output
4	B	B Operand Register - holds the second ALU operand and serves as an accumulator
8	RAD	RAM Address Register - holds cell memory address
1	M	Mark Register - holds mark status of cell
1	C	Carry Register holds ALU carry input

C8-0525

the cells and data from one of the cells. The names and functions of these buses are given in Table 3-6. In addition to the common buses, each cell has a set of rail connections for propagating information between adjacent cells and terminals for passing data between each cell and external devices. These external data ports can be used for cell expansion such as the Direct-Load Memory Feature. The rails to which the cells are connected are part of the response network.

Each cell of an AU may be in either of two major states as determined by the M register in the cell. When the M register is set, the cell is said to be marked, and when clear, the cell is unmarked. The response network deals only with these mark signals.

In addition to the mark signal, several other cell status signals exist, and are listed in Table 3-7. The writing of the cell memory can be conditioned on a selected status signal. Operations of the M (Mark) register can also be conditioned.

The various cell operations and their abilities to be conditionally executed are listed in Table 3-8.

3.5.2 The Response Network

The Response Network is hierarchically organized, using a few types of modules. At the lowest level, each cell has an associated Rail Module. The next level is comprised of Block Response Modules, and above that, Bank Response Modules. In addition to providing an indication (R) when any cell of the system is marked, the Response Network also implements the rails used for cell-to-cell intercommunication. It is the topology of the Response Network that is responsible for the organization of the Associative Processing Cells into blocks and banks.

TABLE 3-6. CELL BUSES

NAME	LINES	FUNCTION
CVB	16	Control Vector - These signals are derived from the AU Program Memory and define the operations to be performed by each cell. (See Table 3-8 for details)
CCB	6	Cell Clocks - These clock signals govern the latching of information into the register of the cells.
DCB	4	Data/Comparand - This bus supplies data which may be used as an operand or comparand by the ALU, or may be written into the cell memory through the ALU.
RAB	8	RAM Address - This bus delivers an address to the cell memories which selects a memory location to be written or read.
RRB	4	Read Response - This bus is used to transfer data or results out of the cells. To avoid bus contention, only the topmost (lowest numbered) cell which is marked has its output register contents placed on the RRB bus.
CIB	4	Cell Internal - This is the only bus of a cell which is unique to that cell, unless the external ports are used to tie cell internal buses together. The internal bus is used to transfer data internally between a cell's ALU and RAM.

C8-0526

TABLE 3-7. CELL STATUS SIGNALS

NAME	DESCRIPTION
1	UNCONDITIONAL - THIS STATUS SIGNAL IS ALWAYS TRUE
M	MARKED - THIS SIGNAL INDICATES WHEN THE CELL IS MARKED
C	CARRY - THIS SIGNAL IS THE VALUE OF THE ALU CARRY INPUT REGISTER
Z	ZERO - THIS SIGNAL IS TRUE WHEN THE ALU OUTPUT IS ZERO
N	NEGATIVE - THIS SIGNAL IS TRUE WHEN THE ALU OUTPUT IS NEGATIVE
P	PROPAGATE - THIS SIGNAL FROM THE RESPONSE NET IS TRUE IF ANY CELL ABOVE THE CELL IN QUESTION IS MARKED
F	FIRST - THIS SIGNAL IS TRUE IF THE CELL IS THE TOPMOST CELL WHICH IS MARKED (P = 0 AND M = 1)
AM	ABOVE MARKED - THIS RAIL INPUT IS THE MARKED STATUS OF THE CELL IMMEDIATELY ABOVE
BM	BELOW MARKED - THIS RAIL INPUT IS THE MARKED STATUS OF THE CELL IMMEDIATELY BELOW

C8-0527

TABLE 3-8. CELL OPERATIONS

CELL COMPONENT	CONTROL SIGNALS	CONDITIONABLE	FUNCTIONS
Arithmetic Logic Unit (ALU)	CCV[0:4]	NO	$\overline{A}$
			$\overline{AB}$
			$\overline{A} \vee B$
			1
			$A \downarrow B$
			$\overline{B}$
			$A \equiv B$
			$A \vee \overline{B}$
			$\overline{A}B$
			$A \oplus B$
			B
			$A \vee B$
			0
			$A\overline{B}$
			AB
			A
			$A - 1$
			$(AB) - 1$
			$(A\overline{B}) - 1$
			$- 1$
			$A + (A \vee \overline{B})$
			$(AB) + (A \vee \overline{B})$
			$A - B - 1$

CS-0526

TABLE 3-8. CELL OPERATIONS (Cont'd)

CELL COMPONENT	CONTROL SIGNALS	CONDITIONABLE	FUNCTIONS
Arithmetic Logic Unit (ALU) Cont'd.	CCV [0:4]	NO	$A \vee \bar{B}$ $A + (A \vee B)$ $A + B$ $(A\bar{B}) + (A \vee B)$ $A \vee B$ $A + A$ $(AB) + A$ $(A\bar{B}) + A$ A
Mark Register (M)	CCV[5:6]	YES	Hold Set Reset Complement
Carry Register (C)	CCV[7:8]	YES	Hold Set Reset Load (from CO)
Memory (RAM)	CCV[9]	YES	Read Write

C8-0526A

TABLE 3-8. CELL OPERATIONS (Cont'd)

CELL COMPONENT	CONTROL SIGNALS	CONDITIONABLE	FUNCTIONS
Internal Bus Source Selection (CIB)	CCV [10:11]	NO	ALU Output RAM Output External Input A (EIA) External Input B (EIB)
Status Condition Selection (S)	CCV [12:15]	NO	1 (Unconditional) M (Mark) C (Carry) Z (+/ALU) N (ALU[0]) P (Propagate) F (First Marked) AM (Above Marked) BM (Below Marked
RR Register (RR)	CCB[2]	YES	Hold Load
B Operand Register (B)	CCB [4]	YES	Hold Load
RAD Register (FAD)	CCB [5]	YES	Hold Load

C8-0528B

Block Response Modules

A Block Response Module serves a block of eight cells and contains eight Rail Modules and one Response Collector Module with the interconnection shown in Figure 3-11. The terminals at the top and bottom of the Block Response Module are for continuation of the rails. The output (BR) which is produced by the Response Collector Module (Figure 3-12), indicates whether any cell of the block is marked. The Response Collector Module is also used to provide a high-speed route for the propagate rail.

Rail Modules

The internal construction of a Rail Module is shown in Figure 3-13. As in the case of the block levels, the terminals on the top and bottom support the intercommunication rails. Each Rail Module has one input from the cell, the contents of its Mark Register (M) and four outputs to its associated cell. These outputs are the status conditions Propagate, P; Above Marked, AM; Below Marked, BM; and First Marked, F.

Bank Response Modules

The Bank Response Module is merely the interconnection of the eight Block Response Modules associated with the blocks of a bank of cells, together with one Response Collector Module. The wiring of a Bank Response Module is shown in Figure 3-14.

Memory Response Module

An Associative Unit contains one Memory Response Module consisting of four Bank Response Modules and other circuitry as shown in Figure 3-15. Signal IP defines the initial condition of the Propagate (P) and Above Marked (AM) Rails. The signal NP inhibits the Propagate Rail between the banks of cells. This allows four cells, one in each bank, to be supplied with different data, such as would be done during a fast overlay of the cell memories.

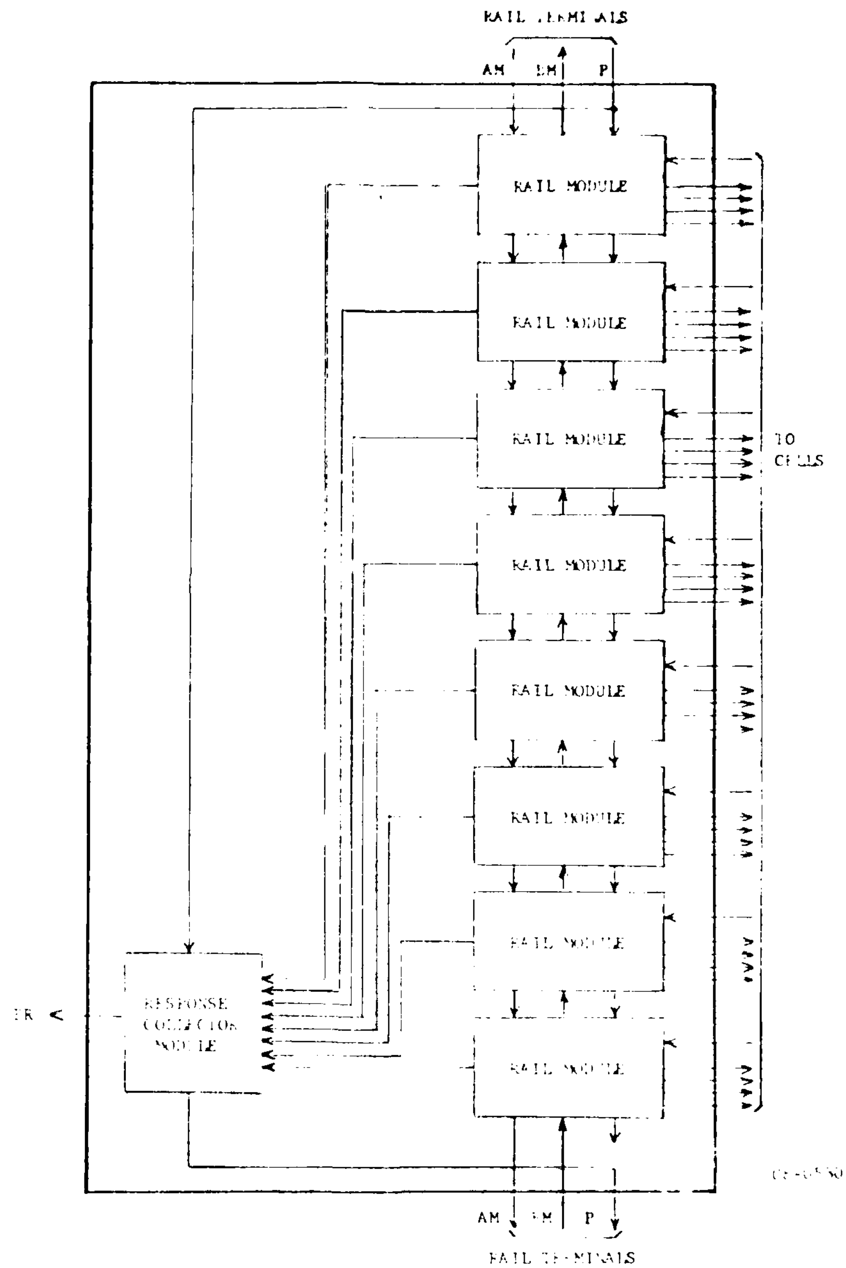


Figure 3-11. Block Response Module

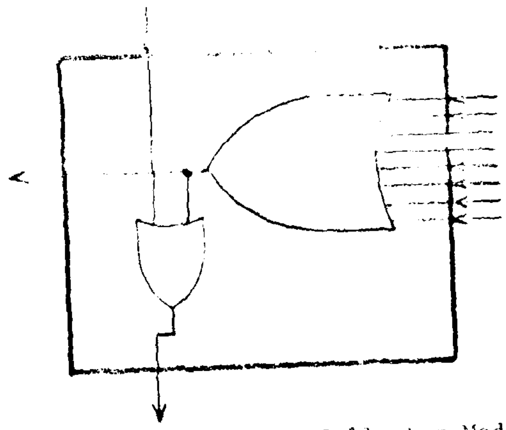


Figure 3-12. Response Collector Module

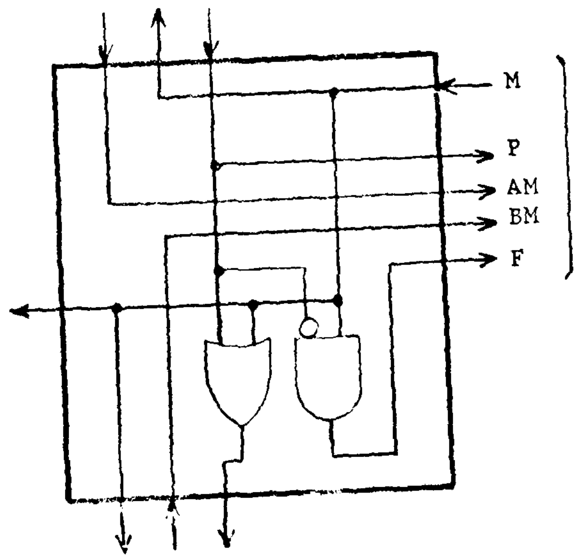


Figure 3-13. Rail Module

C8-0531

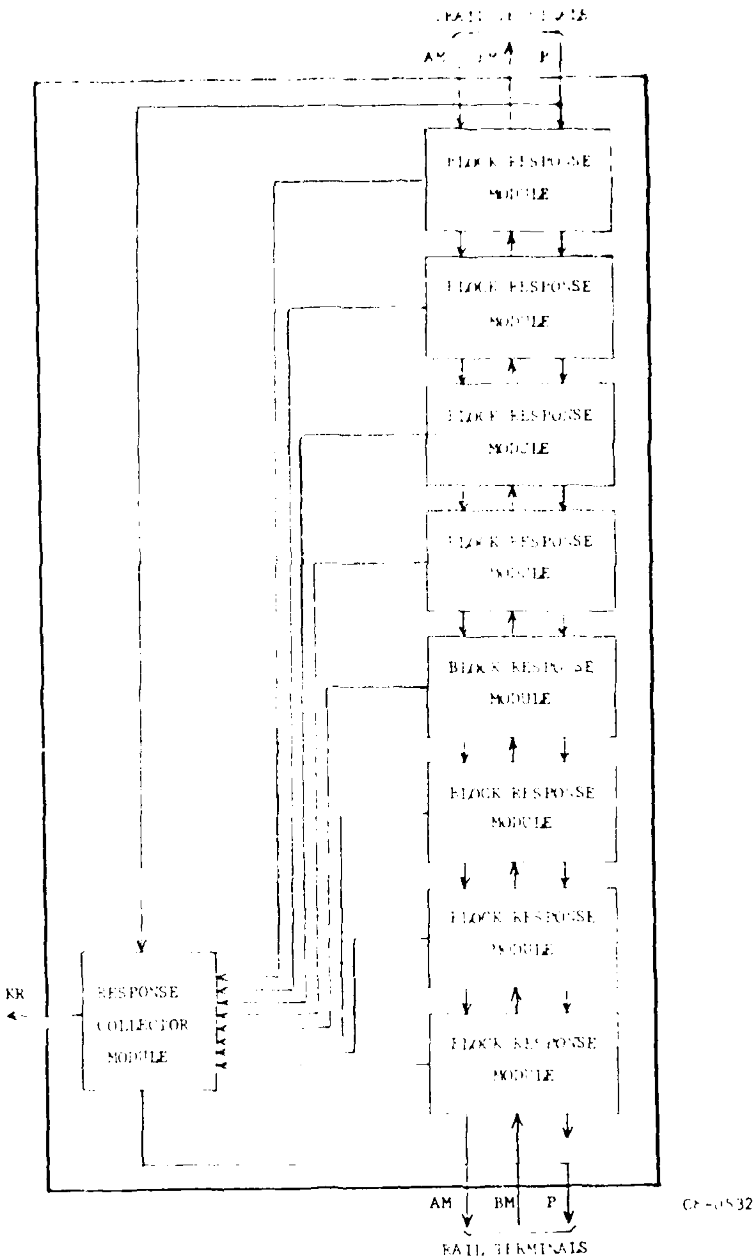


Figure 3-14. Bank Response Module

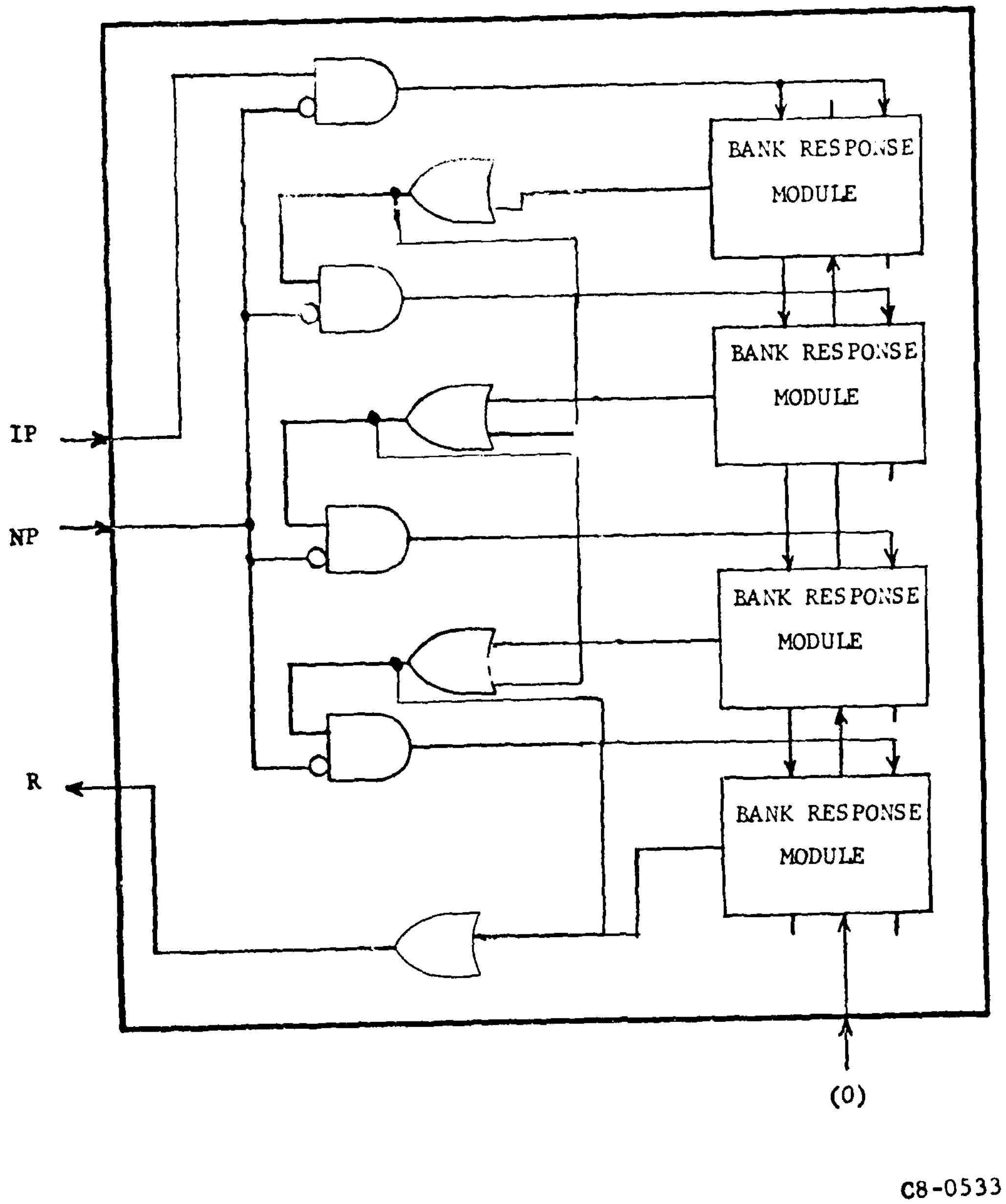


Figure 3-15. Memory Response Module

3.5.3 The Controller

A block diagram showing the functional organization of the Controller is shown in Figure 3-16.

Interface Connections

Figure 3-17 shows some of the Controller registers and how they are connected to the Interface Unit. Input data and search requests are received by the Controller in a queue or FIFO having a capacity for 16 16-bit quantities. The input clock causes the value on the data input lines to be placed at the end of the queue if the queue is not full. The control clock causes the value on the control input lines to be loaded into a control register, CTL. The contents of CTL are used to determine the mode in which the Controller operates as described in the next section. The control information also selects the status condition to be used to signal an interrupt request.

The output clock informs the Controller that the contents of the data output buffer (BUF) have been accepted by the host processor. The output clock also clears an "output ready" status bit in the Controller.

The status of the Associative Unit is reflected in the contents of a status register, STAT. Represented status conditions are specified in Table 3-9. Any one of the status bits can be selected, as specified through CTL, to serve as an interrupt request signal.

Operational Modes

The Controller has four operational modes as listed in Table 3-10. Switching between modes is governed by the host processor through a field in the control register, CTL. A transition from the Operate Mode to the Wait Mode can, however, be caused by the Controller itself.

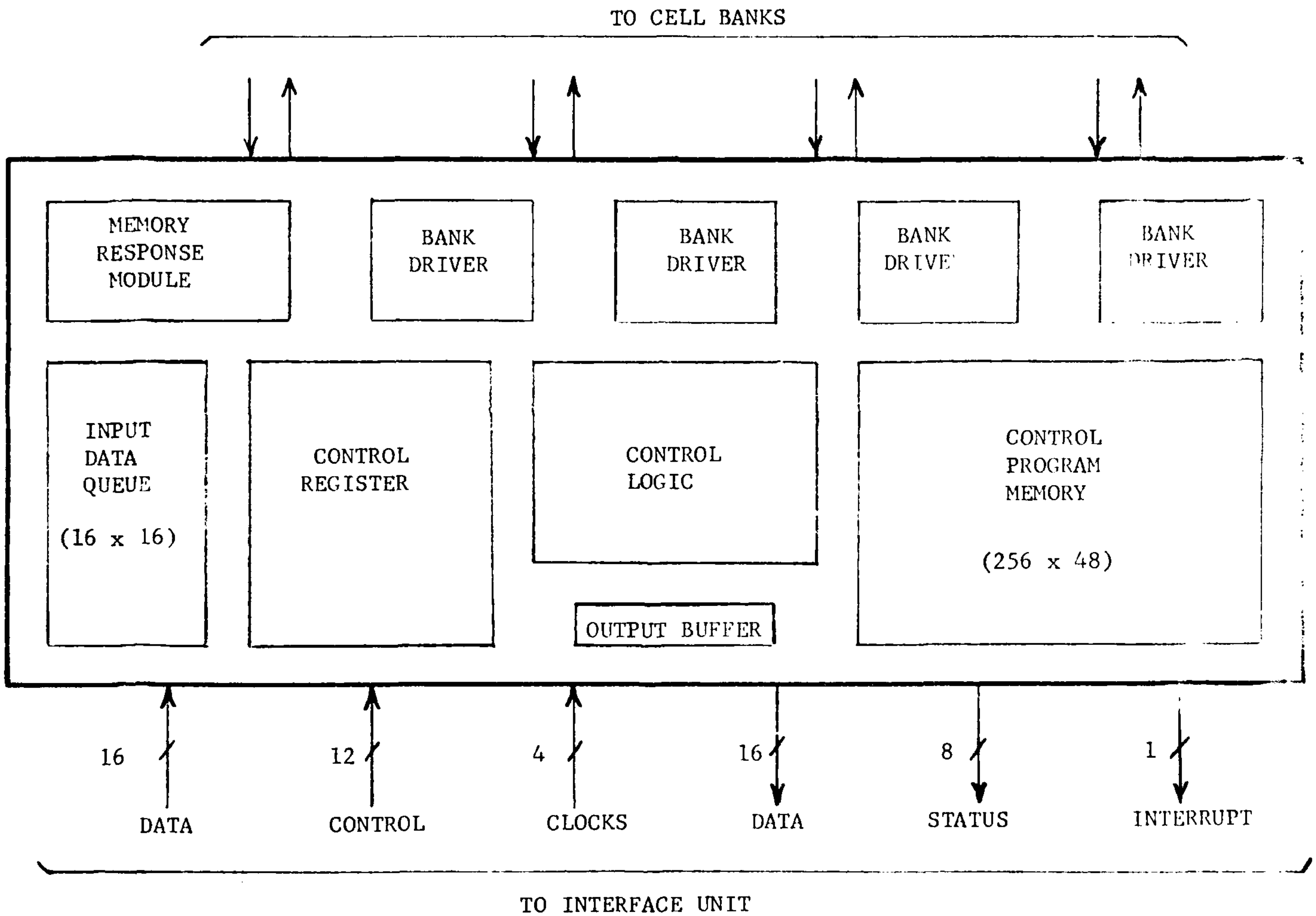
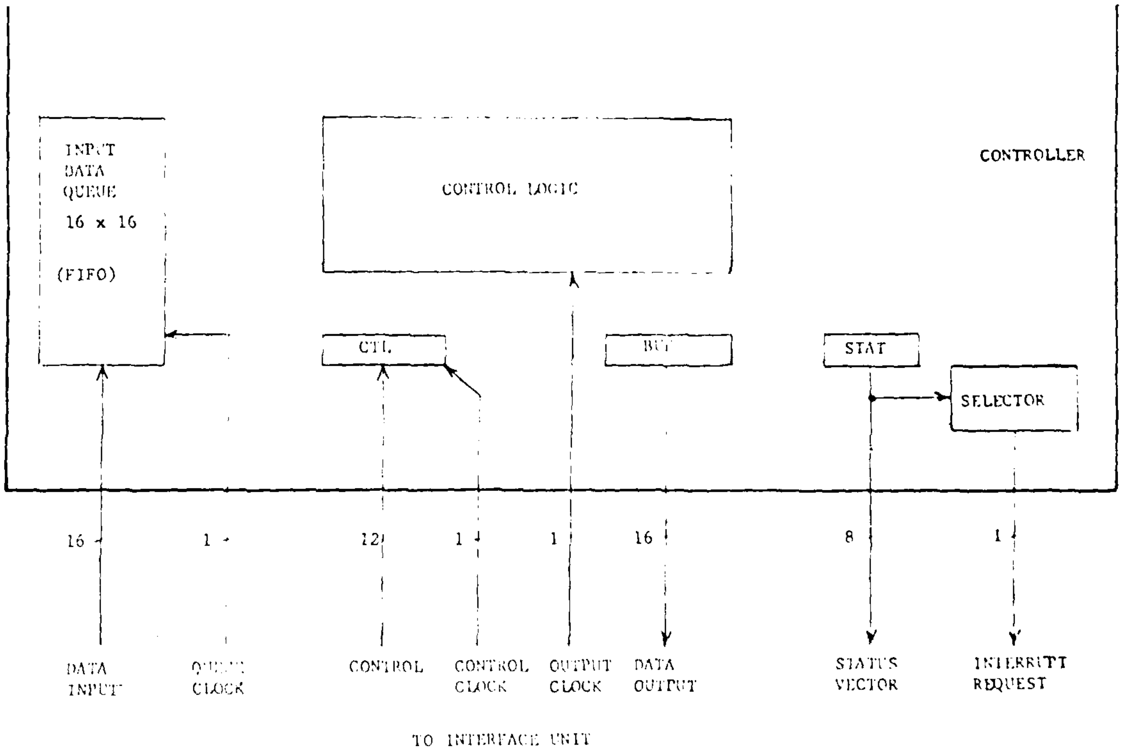


Figure 3-16. Controller Organization

C8-0534



CF-0535

Figure 3-17. Controller-Interface Connections

TABLE 3-9. STATUS CONDITIONS

Input Queue Full
Input Queue Empty
Output Ready
Response
Wait Mode
Counter A Zero
Counter B Zero

C8-0536

TABLE 3-10. OPERATIONAL MODES

WAIT	-	The AU is quiescent.
OPERATE	-	The AU is under control of its own program.
INTERROGATE	-	The AU Controller Registers and Program Memory can be interrogated by the host processor.
REPROGRAM	-	The AU Program Memory can be overlaid by the host processor.

C8-0537

In the Wait Mode, the AU is essentially in a halt state. When the mode is changed to the Operate Mode by the host processor, the first quantity placed in the Input Data Queue (FIFO) is interpreted to be the address of a program in the Program Memory, and is loaded into the Program Memory Address Register (PAD). Thereafter, the AU is under control of its own program until the mode is respecified by the host through the control port, or the program returns the Controller to the Wait Mode. Once the program memory address register is loaded on entering the Operate Mode, the subsequent quantities may be interpreted by the Controller as either data or further specifications on the operations to be performed.

When the Interrogate Mode is entered, the contents of various registers of the Controller and the contents of the Program Memory are placed in the output buffer register (BUF) 16 bits at a time. The sequence of interrogating the Controller is illustrated in Table 3-11. The interrogation is progressed each time the Output Clock is pulsed, and terminates when another mode is specified by the host. The Controller registers are listed in Table 3-12.

In the Reprogram Mode, the program memory can be overlaid by the host processor through the Input Data Queue (FIFO). On first entering this mode, the Input Data Queue is cleared. Following this, the first entry in the queue is loaded into the Program Memory Address Register (PAD) to specify the first word in the memory which will receive new program information. Subsequent information from the queue is loaded into sequential locations in the program memory. A reprogramming example starting at memory location I is shown in Table 3-13. The Controller will remain in this mode until another mode is specified.

<u>Controller Register Operations</u>

Each of the primary registers of the Controller is capable of a set of register transfer operations as defined in Table 3-14. The conditional loading of the program memory address register is used for branching. A field of the program memory word can be used as a branch condition. An instruction can be repeated until the specified condition occurs by having

TABLE 3-11. INTERROGATION SEQUENCE

SEQUENCE NUMBER	INFORMATION SOURCE	
1	Program Memory Address Register	PAD
2	Cell Memories Address Register and Status Register	CAR, STAT
3	Count Registers	CTA, CTB
4	Memory Address 0, Left	MP[0; 0:15]
5	Memory Address 0, Center	MP[0;16:31]
6	Memory Address 0, Right	MP[0;32:47]
7	Memory Address 1, Left	MP[1; 0:15]
⋮	⋮	

C8-0538

TABLE 3-12. CONTROLLER REGISTERS

NAME	LENGTH	USE
PAD	8	Program Memory Address
RAR	8	Cell Memory Address Broadcast to All Cells
STAT	8	Controller Status Conditions
CTA	8	Counter Used by Programs
CTB	8	Counters Used by Programs
CTL	12	Control Input From Host Processor
BUF	16	Data Output Buffer Register

C8-0539

TABLE 3-13. REPROGRAMMING SEQUENCE

Clear Input Data Queue	
Load Program Memory Address with I	PAD I
Load Memory Address I, Left	MP [I; 0:15]
Load Memory Address I, Center	MP [I; 16:31]
Load Memory Address I, Right	MP [I; 32:47]
Load Memory Address I+1, Left	MP [I+1; 0:15]
•	
•	
•	

C8-0540

TABLE 3-14. CONTROLLER REGISTER OPERATIONS

PAD	Program Memory Address Register
	Hold on condition false.
	Load from Program Memory Data Field on condition TRUE.
	Load from Program Memory Data Field on condition FALSE.
	Increment
	*Clear
	*Load from Input Data Queue
RAR	Cell Memory Address Register
	Hold
	Load from Counter A
	Load from Input Data Queue
	Increment
	Decrement
	Clear
CTA	Counter A
	Hold
	Load from Program Memory Data Field
	Load from Input Data Queue
	Decrement
	*Clear
CTB	Counter B
	Same as Counter A
BUF	Output Buffer Register
	Hold
	Load first 4 bits from RRB bus
	Load next 4 bits from RRB bus
	Clear

C8-0541

TABLE 3-14. CONTROLLER REGISTER OPERATIONS (CONT'D)

BUF	*Load from Pad
	*Load from RAR and STAT
	*Load from CTA and CTB
	*Load from Left Program Memory
	*Load from Center Program Memory
	*Load from Right Program Memory
	(* supports Interrogate and Reprogram Modes, not available to programmer.)

C8-0541A

the address register hold its contents as long as the condition is false. Alternatively, a conditional jump can be programmed by having the address register loaded from a field in the program memory if the specified condition is met.

Bank Drivers

The data and control information is transmitted from the Controller to the banks of Associative Processing Cells through a set of Bank Drivers. These drivers are identical except for the way a 4-bit datum is obtained for the DCB bus of each bank.

The 16-bit quantity on the front of the Input Data Queue can be broadcast to all cells in 4-bit segments from right to left in a series of AU cycles. In this case, the DCB input to every cell of the Associative Unit is the same in a given cycle. Alternatively, the DCB bus for Cell Bank 0 can be driven with the first (bits 0-3) 4-bit segment from the front of the queue, and simultaneously, Banks 1, 2, and 3 are driven with the second, third, and fourth segments, respectively.

The Bank Drivers also include receivers for the RRB bus of each bank.

Program Memory Instruction Fields

Each 48-bit word of the Program Memory can be interpreted as a horizontal microinstruction. Only one of the fields has two uses. This field can be used to specify a control vector to be broadcast to all cells, or can be used for data such as a branch address or an initial value for control registers. The uses of the various fields are listed in Table 3-15.

AD-A082 324 CONTROL DATA CORP MINNEAPOLIS MINN F/G 9/2
APPLICATIONS OF A RECONFIGURABLE ARRAY OF FLEXIBLE PROCESSORS I--ETC(U)
DEC 79 W R CYRE F30602-78-C-0065
UNCLASSIFIED RADC-TR-79-313 NL

2 OF 2
AD A082324

TABLE 3-15. PROGRAM MEMORY INSTRUCTION FIELDS

Bits	Functions
16	Cell Control Vector or Data
1	Specifies Data Field
3	PAD operation specification
3	RAR operation specification
3	CTA operation specification
3	CTB operation specification
2	BUF operation specification
1	Remove front of FIFO
1	Set output ready condition
1	Enter Wait Mode
1	Select DCB source and P-Rail Bank Inhibit
1	P-Rail initial value
3	Branch Condition Select
3	Cell Clocks
42	

C8-0542

4.0 A DEMONSTRATION DOCUMENT RETRIEVAL SYSTEM

This section describes a demonstration document retrieval system developed at Control Data to evaluate the application of a Reconfigurable Array of Flexible Processors to document retrieval for assessment of foreign technology. The hardware support for the demonstration system was a Laboratory Array of Flexible Processors to which the Exploratory Development Model Associative Unit was attached. The hardware configuration of the Laboratory Array is shown in Figure 4-1.

4.1 GENERAL DESCRIPTION

The basic concept on which the demonstration system was built employed the model of Section 2.2 and is illustrated in Figure 4-2. As shown, the system concept has six functional elements. The allocation of the Laboratory Array hardware to these functions is described in Figure 4-3. The user or intelligence analyst interacts with the Command Entry and Display Elements, which consists of a character-oriented CRT display and keyboard (DISP CRT). The second element of the concept (Command Analysis and Display Preparation) analyzes the user input commands, breaking them down into lists of simpler functions, and formats the results of a search for display. The Command Analysis and Display Preparation (CADP) function is supported by the CDC 1700 general purpose computer of the Laboratory Array as indicated in Figure 4-3. The 1700 computer also delivers copies of the input commands and output displays with other intermediate data to a line printer (PTR) for performance analysis.

The File Management (FM) functional element maintains the data base files and transfers them to the Search Execution (SE) element or to the display as required. Both the File Management and Data Base (DB) elements are supported by the Reconfigurable Array of Flexible Processors in the Laboratory Array. The Flexible Processors (FP) support the transfer of files, while the data base is stored in the disk and RAM memories. During

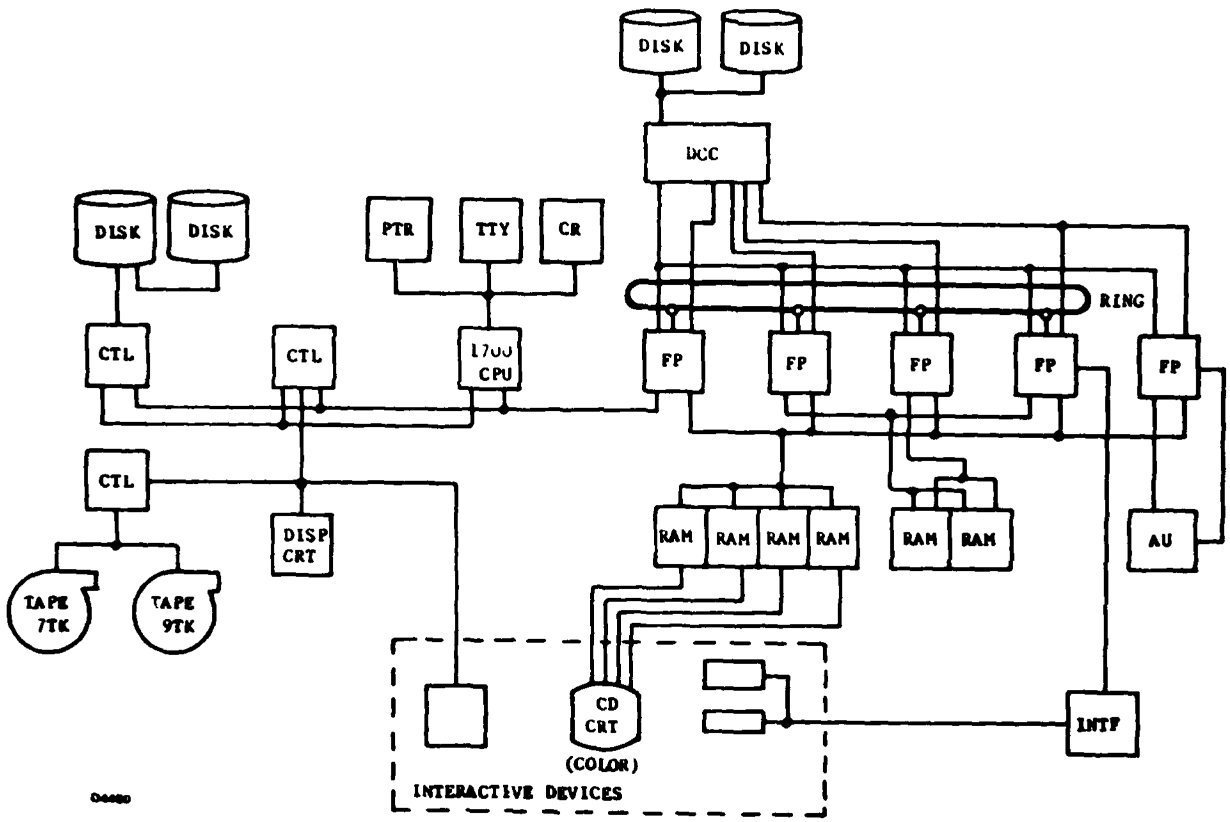


Figure 4-1. The Laboratory Array

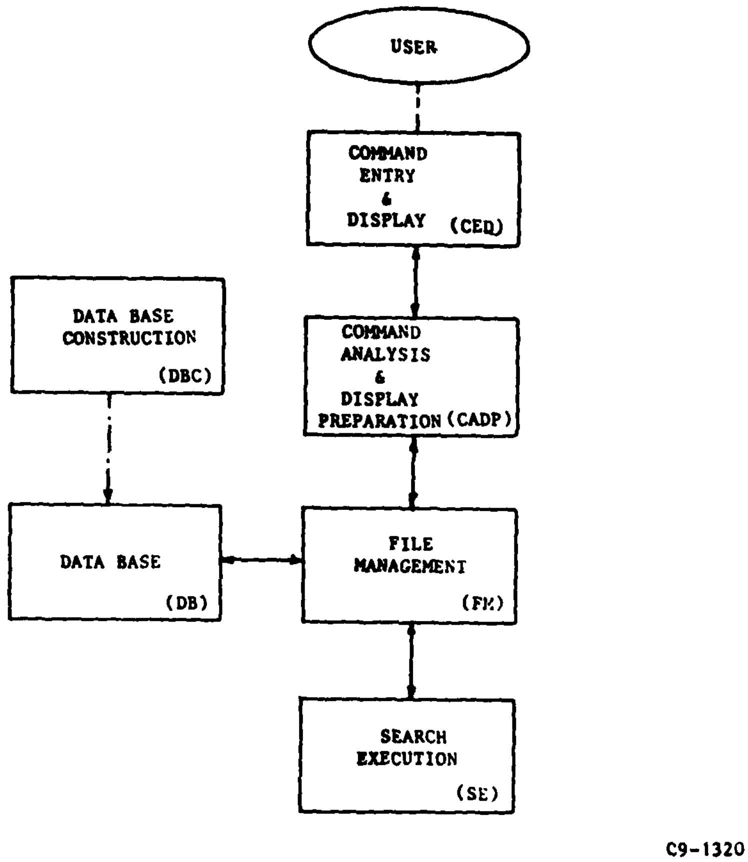


Figure 4-2. Demonstration System Concept

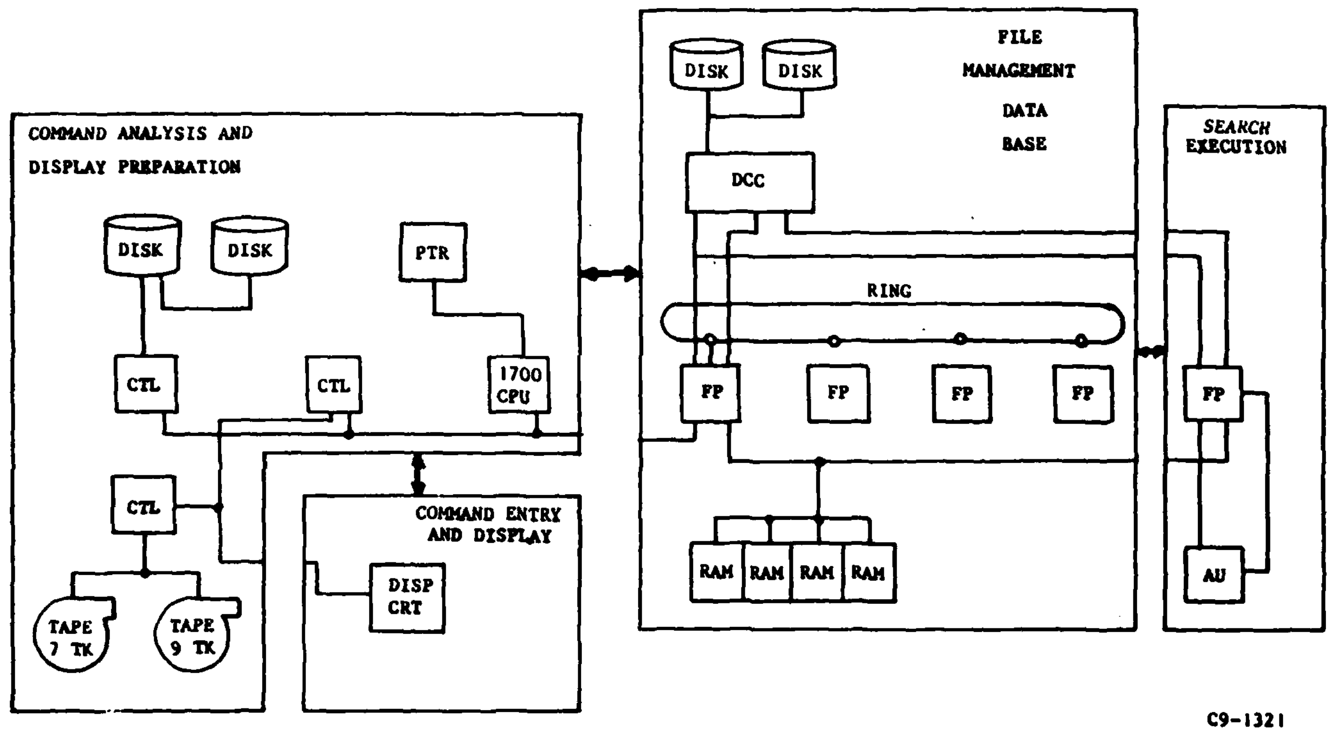


Figure 4-3. Functional Allocation of the Laboratory Array

operation, the SEGMENT TABLE, DICTIONARY, and OCCURRENCE files of the Experimental SA Data Base are stored in the RAM memory for rapid access, while the DOCUMENT file resides in the disk units connected with the Data Channel Controller (DCC). With a larger data base, all data base files would reside in disk storage.

The Search Execution element, which performs the actual search operations on occurrence lists, is supported by a Flexible Processor and the Exploratory Development Model Associative Unit (AU).

One element, the Data Base Construction Element (DBC), is not supported by the Laboratory Array. The data base is constructed off-line by FORTRAN software running on a CDC 6400 general-purpose computer.

4.2 DEMONSTRATION CAPABILITIES

The demonstration system allows four basic types of commands for experiments in document retrieval. These commands initiate searches, display of documents or occurrence lists, and display of the history of searching for a terminal session.

4.2.1 Search Commands

A search command consists of an identifier and a search expression. In the demonstration system, only binary search operations for two terms are allowed. As described shortly, highly complex search expressions can be synthesized by sequences of allowed commands. A search term can be any alphanumeric string of up to 16 characters with no embedded blanks, or can be a search command identifier.

Four search operators have been implemented, and they are OR, AND, NOT, and ADJ (adjacent). The OR operation produces a list of document numbers, word number pairs. If one term of the search expression has no occurrences (is not in the dictionary), then the occurrence list of the other term is returned as a result. When both terms have nonvacuous occurrence lists, the list for the righthand term is loaded in the Associative Unit (AU) and each document number of the occurrence list for the left term is compared with the document numbers in the AU. Where no match on document numbers occurs, the document number, word number pair from the left term's list, is added to a result list. After all entries in the left term's list have been examined, the occurrence list for the righthand term is appended to the partial result list to form the final result list. Since the result list has the properties of an occurrence list, it can be used later as an operand for a search. The NOT search operation is performed in the same manner as the OR operation, except the occurrence list for the right term is not included in the result. Examples of the result lists for (CAN or BE) and (CAN not BE) appear in Table 4-1.

The AND and ADJ search operations are processed somewhat differently. Again the right term list is loaded into the Associative Unit. For the AND operation, each occurrence for the left term is compared with the Associative Unit contents for matches on document numbers. When a document number match occurs, the occurrence from the right term's list is added to the result list. With the ADJ operation, one is added to the word number part of each occurrence of the left term before comparison. Both the document number and word numbers are compared for a match, and when a match occurs, the entry from the right term's list is added to the result list.

To illustrate this discussion two examples of searches from a demonstration session are reproduced in Table 4-2. In these examples the user was searching with the expression ((HORMONE or HORMONES) and BRAIN), by decomposing it into two binary operations. As indicated, the result list from one search can be used as an operand in a later command. Thus, searches of any complexity can be performed with the demonstration system. In addition to giving the occurrence list lengths of the operands and the result, the system returns an indication of the times required to perform the search. The AU LOAD time is the time required to load the Associative Unit with the occurrence list for the righthand term of the search expression. The AU SEARCH time shows how long it took to compare all entries in the left term's occurrence list.

TABLE 4-1. RESULT LIST EXAMPLES

LEFT TERM		RIGHT TERM		RESULT LISTS							
CAN		BE		CAN or BE		CAN not BE		CAN and BE		CAN adj BE	
39 Occurrences		44 Occurrences		66 Occurrences		22 Occurrences		19 Occurrences		10 Occurrences	
DN	WN	DN	WN	DN	WN	DN	WN	DN	WN	DN	WN
1	145	2	144	1	145	1	145	12	152	12	152
5	153	7	136	1	153	1	153	23	154	23	154
12	151	11	159	17	132	17	132	30	146	33	132
17	132	12	152	26	149	26	149	33	132	35	147
23	153	20	156	34	130	34	130	35	135	46	160
26	149	23	154	41	132	41	132	35	147	50	140
30	144	25	146	45	138	45	138	46	160	113	137
33	131	30	146	48	156	48	156	50	140	134	142
34	130	31	135	52	133	52	133	65	141	137	146
35	146	33	132	57	143	57	143	78	152	154	134
41	132	35	135	62	137	62	137	97	148		
45	138	35	147	69	149	69	149	101	24		
46	159	38	138	75	132	75	132	113	137		
48	156	38	159	90	138	90	138	124	148		
50	139	40	144	102	142	102	142	124	154		
52	133	43	154	102	153	102	153	134	142		
57	143	46	160	115	156	115	156	137	147		
62	137	50	140	132	136	132	136	154	134		
65	139	64	151	148	131	148	131	176	144		
69	149	65	141	150	153	150	153				
75	132	71	147	153	132	172	144				
75	151	78	152	172	144						
90	138	84	132	2	144						
97	146	84	155	7	136						
101	39	86	150	11	159						
102	142	87	155	12	152						
102	153	93	135	20	156						
113	136	97	148	23	154						
115	156	101	24	25	146						
124	153	113	137	30	146						
132	136	116	152	31	135						
134	141	123	158	33	132						
137	146	124	148	35	135						
148	131	124	154	35	135						
150	153	129	150	38	138						
153	132	129	157	38	159						
154	133	134	142	40	144						
172	144	137	147	43	154						
176	133	140	159	46	160						
		147	149	50	140						
		154	134	64	151						
		161	144	65	141						
		176	143	71	147						
		188	144	78	152						
				.	.						
				.	.						
				188	144						

TABLE 4-2. TWO SEARCH EXAMPLES

```
@c = hormone  or hormones
@C=HORMONE           .OR.  HORMONES
NUMBER OF OCCURRENCES-A=  / B=  5 RESULT=  5
AU LOAD    45, AU SEARCH=    2 (IN MICROSECONDS)


@d = @c and brain
@D=@C                AND.  BRAIN
NUMBER OF OCCURRENCES-A=  5 B=  8 RESULT=  2
AU LOAD=   42, AU SEARCH=   11 (IN MICROSECONDS)


       NOTE:  User command    -  lower case

              System response -  upper case
```

C9-1318

4.2.2 Read Commands

The function of a search command is to cause a search operation to be performed on the specified terms. A search command form called the read command allows the user to read and display the occurrence list for any term or search result. The displays for this command on the result of the examples of Table 4-2 and for a term are reproduced in Table 4-3. For the artificial term @d, both occurrences are in document number 50, entitled "Interactions Between Hormones and Nerve Tissue." The "SECTION S" indicates that the word brain is found in the summary section of the document, and the quantities in parentheses are the document number, word number pairs as they are represented in the data base. Note that the SECTION information is encoded into the word number representation of an occurrence. The second example of Table 4-3 shows the display for a read command with respect to the term "BRAIN."

4.2.3 Document Display Commands

A third command form provided in the demonstration document retrieval system permits the user to display the texts of documents by specifying the document numbers. An example for this command appears as Table 4-4. Note that the two occurrences in the result for the example of Table 4-2 are due to the two occurrences of "BRAIN," though "hormones" has three occurrences.

4.2.4 Search History Commands

The final command form permits the user to display a summary of the history of a search session. This history of search commands is useful in composing complex search expressions. An example is given in Table 4-5.

4.3 DEMONSTRATION SYSTEM PERFORMANCE

As a basis for evaluating the performance of the Associative Unit in executing search operations, over 250 searches were performed. Two performance measures were selected: the time to load the Associative Unit

TABLE 4-3. READ EXAMPLES

```
read  @d
THE TERM- @D                HAS    2 OCCURRENCES
DOCUMENT  50, SECTION S, WORD 19 ( 50 147)
DOCUMENT  50, SECTION S, WORD 31 ( 50 159)


read brain
THE TERM- BRAIN             HAS    8 OCCURRENCES
DOCUMENT   6, SECTION S, WORD  7 (  6 135)
DOCUMENT  50, SECTION S, WORD 19 ( 50 147)
DOCUMENT  50, SECTION S, WORD 31 ( 50 159)
DOCUMENT  57, SECTION S, WORD 10 ( 57 138)
DOCUMENT  88, SECTION T, WORD  8 ( 88  16)
DOCUMENT  88, SECTION S, WORD 10 ( 88 138)
DOCUMENT 141, SECTION S, WORD 31 (141 159)
DOCUMENT 158, SECTION T, WORD  5 (158  13)
```

C9-1315

TABLE 4-4. DOCUMENT DISPLAY EXAMPLE

```
doc 50
50 A 7607MCEWEN
50 T INTERACTIONS BETWEEN HORMONES AND NERVE TISSUE
50 S STEROID HORMONES SECRETED BY THE GONADS AND THE ADRENAL CORTEX CAN BE
50 S    TRACED
50 S TO TARGET CELLS IN THE BRAIN. IN THE NEWBORN ANIMAL THE SEX HORMONES HELP
50 S TO LAY DOWN BRAIN CIRCUITS THAT CONTROL LATER BEHAVIOR
```

C9-1316

TABLE 4-5. SEARCH HISTORY EXAMPLE

```
dh1
@A=THEORIES          .OR.  THEREBY
@B=TIN               .OR.  @A
@C=THINK             .OR.  @B
@D=BRAIN             AND.  CELLS
@E=OTHER             ADJ.  ACT
@F=NEW               NOT.  ANY
```

C9-1317

with the occurrence list of the righthand search term, and the time to compare each entry of the occurrence list of the lefthand term. These performance measurements are plotted in Figures 4-4 through 4-9.

The Associative Unit loading time for the demonstration system is plotted in Figure 4-4 as a function of the number of occurrences for the righthand search term. A broken curve connects the data for low numbers of occurrences to clarify the relationship. This peculiar behavior is due to the Flexible Processor routine which zero-fills an occurrence list to a multiple of 32. This zero-filled routine used in the demonstration was slower than the loading routine. In operational practice, the zero-fill would be achieved by preclearing the associative memory. This would require only a couple of microseconds per column. For long occurrence lists, the load rate is 675 nanoseconds per occurrence. For the occurrence lists found to occur with the FTD searches, an average list of 2905 occurrences could be loaded in about two milliseconds if 16-bit document number, word number pairs were used. Since the occurrence list entries would need to be on the order of 48 bits, the average load time would be closer to 6 milliseconds.

The times required by the Associative Unit to compare the occurrences of the lefthand search terms with the Associative Unit contents are segregated by the type of operation in Figures 4-5 through 4-8. As can be seen in the figures, the search time is not strongly affected by the type of operation.

Figure 4-9 shows the effect of the number of occurrences for the righthand term on the Associative Unit search time with respect to the OR operation. The numbers next to the curves indicate the number of left term occurrences. These characteristics are approximated by Equation 4-1 below where T is the search time in microseconds, L_L is the number of occurrences for the lefthand term, and L_R is the number of occurrences for the righthand search term.

$$T = 0.3 \times L_L \times L_R^{0.6} \qquad 4\text{-}1)$$

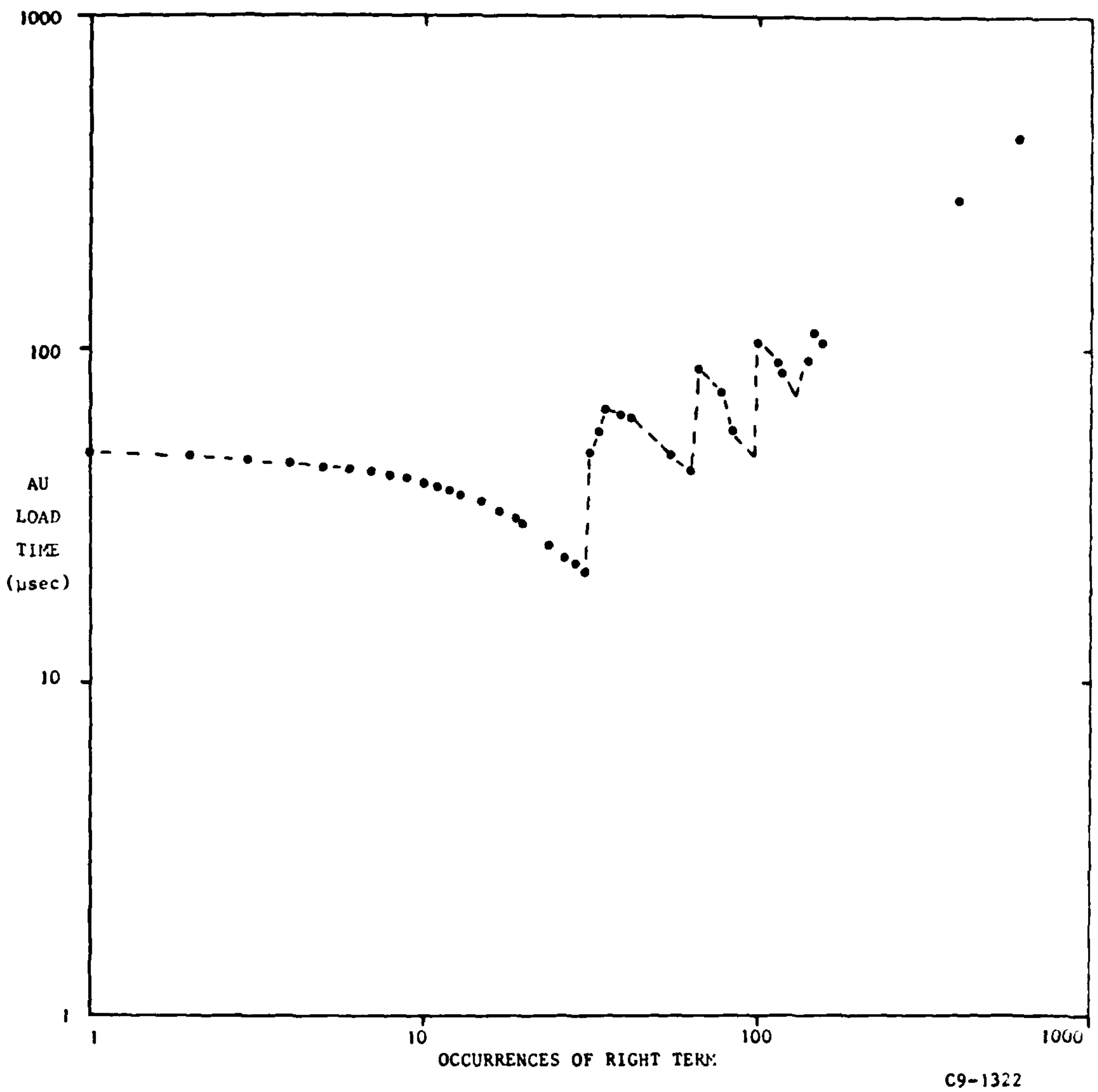


Figure 4-4. AU Loading Performance

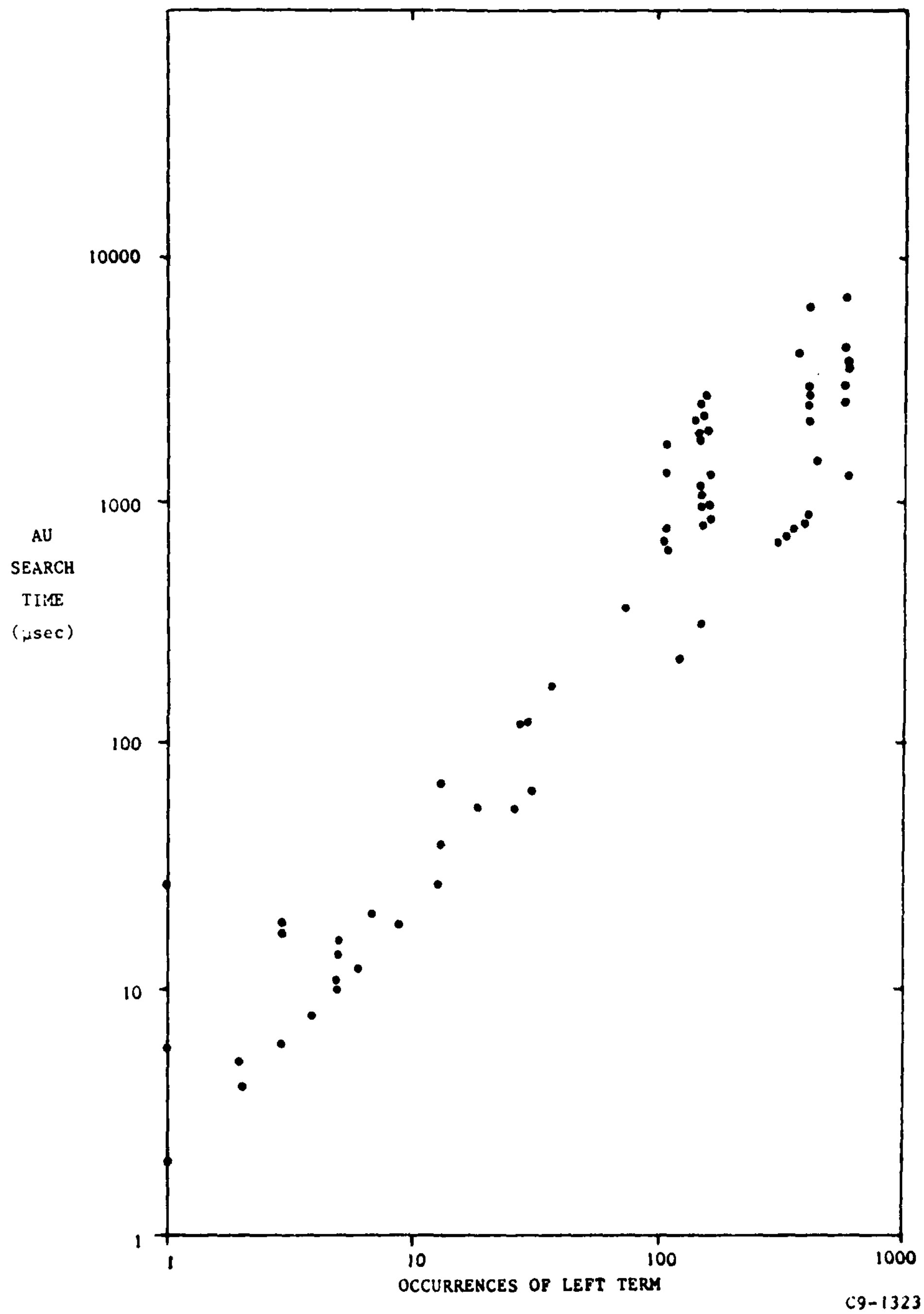


Figure 4-5. AU Search Performance (OR Operation)

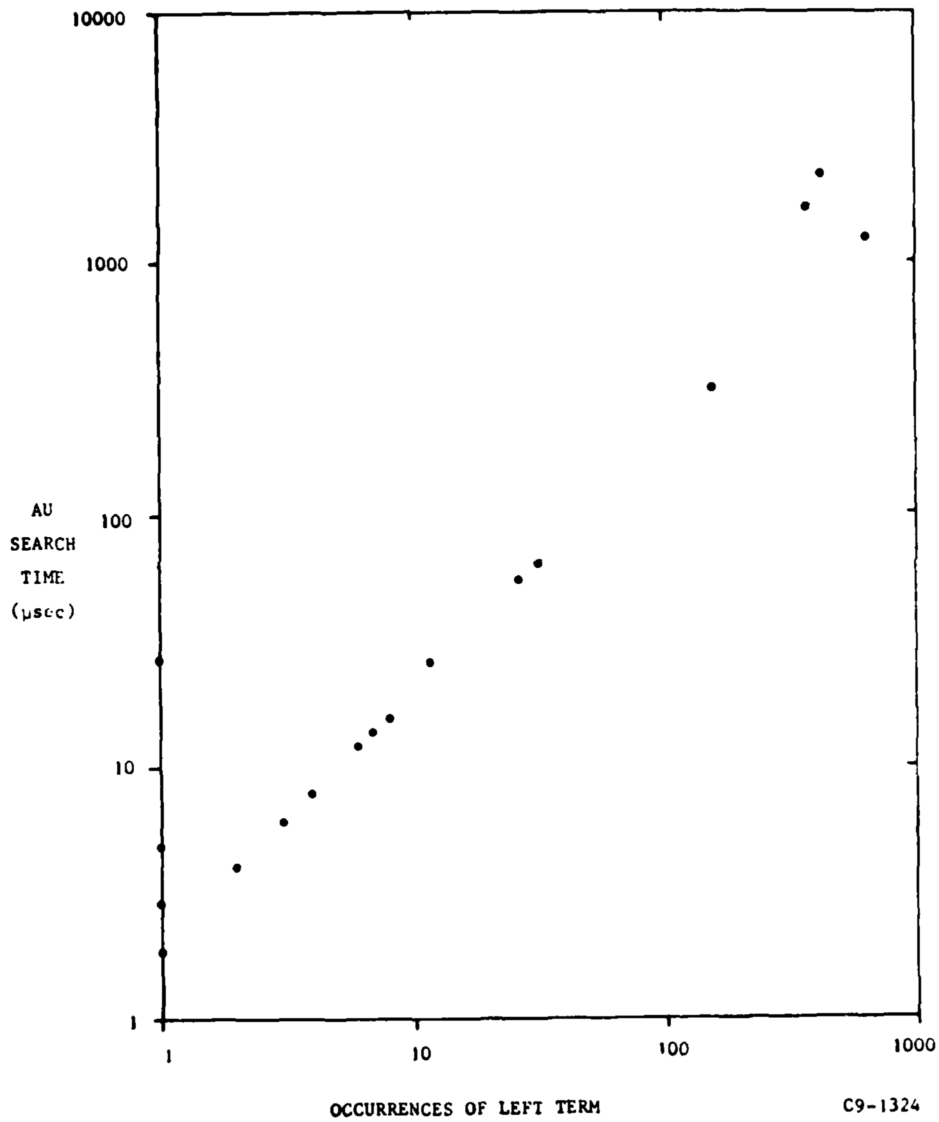


Figure 4-6. AU Search Performance (NOT Operation)

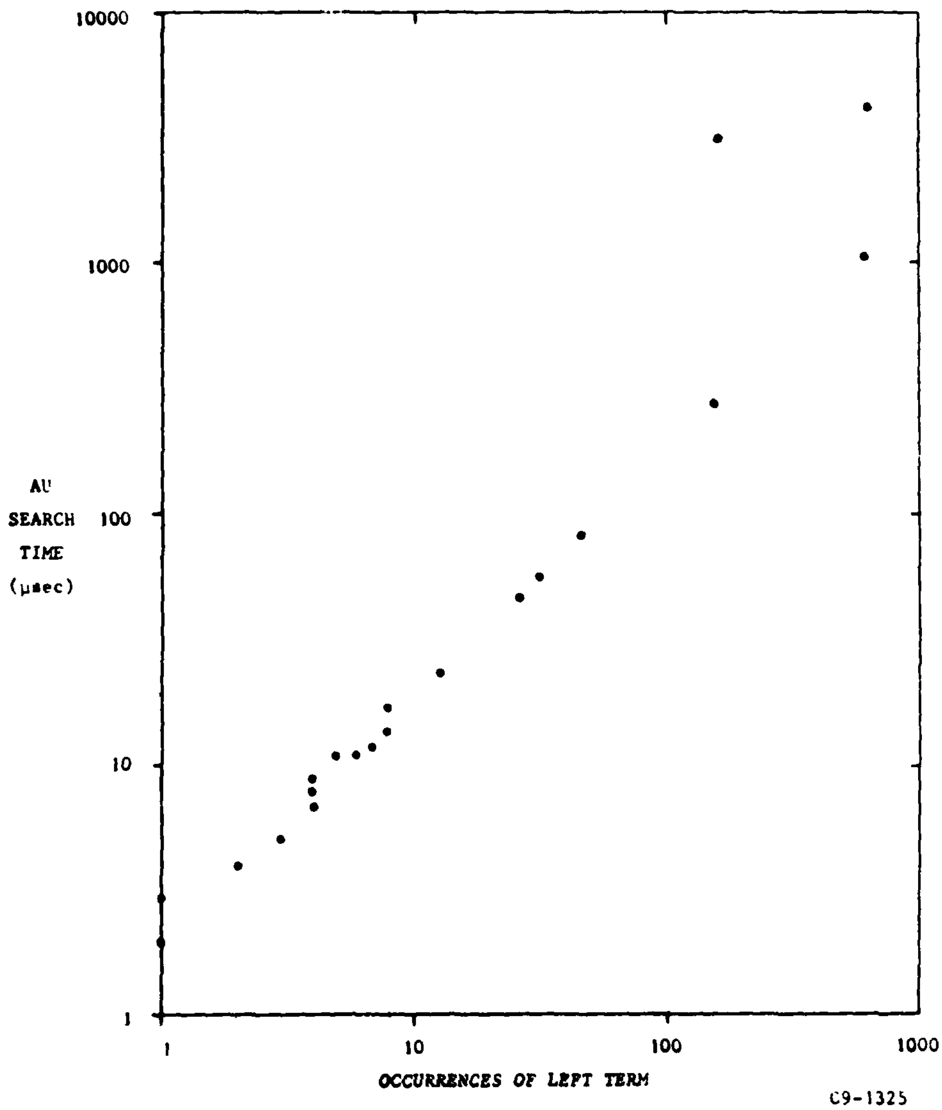


Figure 4-7. AU Search Performance (AND Operation)

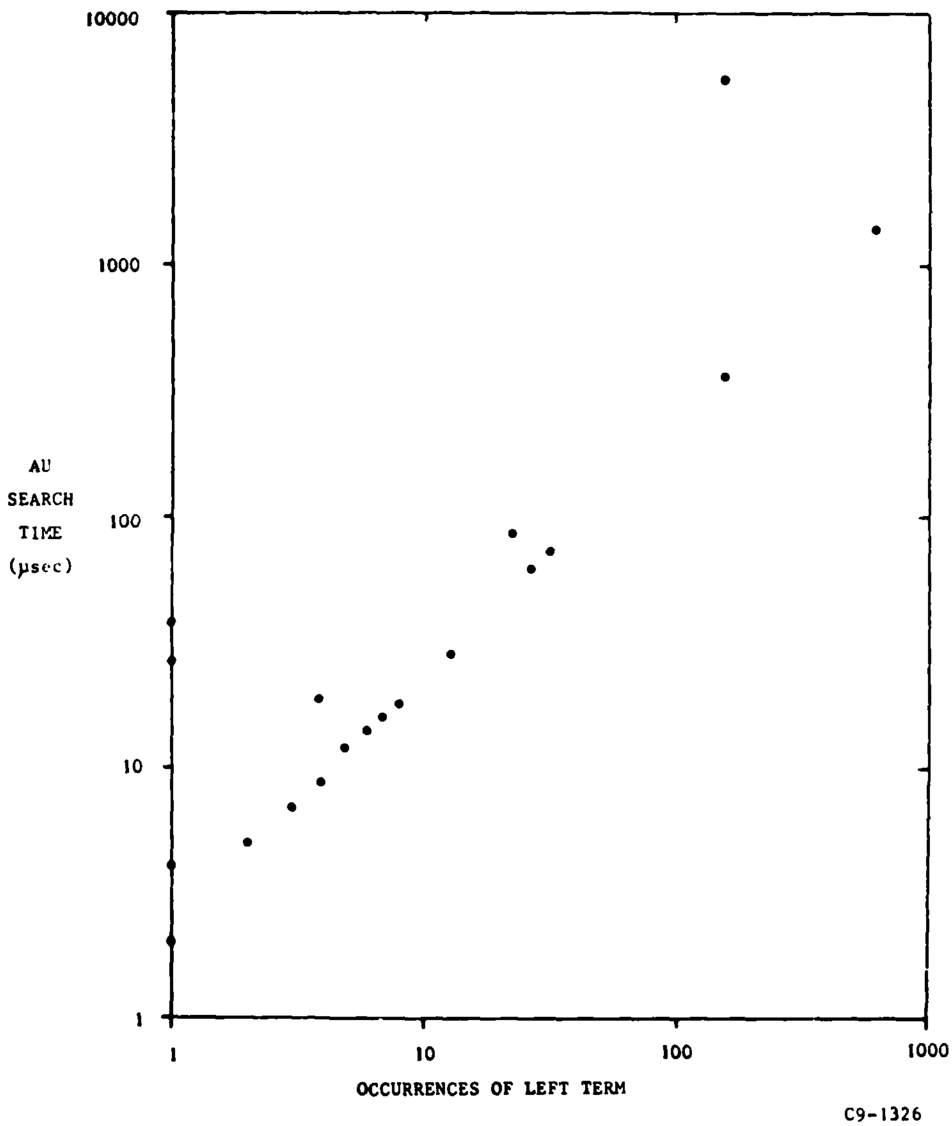


Figure 4-8. AU Search Performance (ADJ Operation)

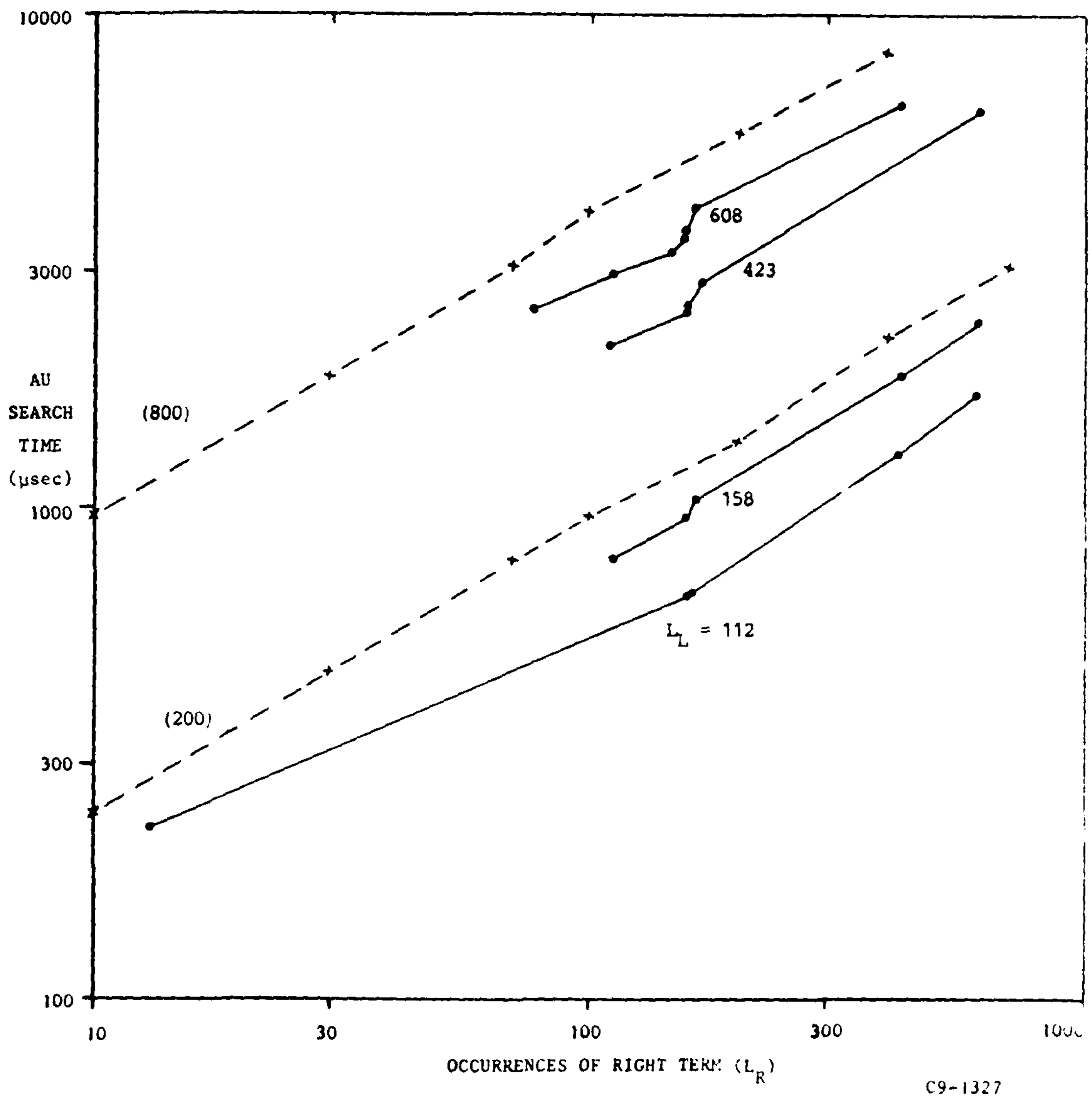


Figure 4-9. AU Search Performance (OR Operations)

TABLE 4-6. RECONFIGURABLE ARRAY PERFORMANCE COMPARISON

	32 - CELL ASSOCIATIVE UNIT	256 - CELL ASSOCIATIVE UNIT	
COMMAND ANALYSIS TIME	2	2	MSEC.
FILE RETRIEVAL TIME	68	68	MSEC.
AU LOAD TIME	6	6	MSEC.
AU SEARCH TIME	312	39	MSEC.
TOTAL	388	115	MSEC.
PERFORMANCE RATIO	66	223	

NOTES: AU search and load times assume operands of 2905 occurrences of 48 - bits each. Performance ratio based on existing system average search time of 25600 milliseconds (see Section 2.1.)

C9-1328

5.0 CONCLUSIONS

The results of this study program indicate considerable promise of dramatic performance improvement with a Reconfigurable Array in intelligence information retrieval. The array with an associative unit is well suited for the basic document retrieval tasks: file management, data base update, and search. In the search function, the reconfigurable array approach appears to be 200 times faster than the current system. Further study with a larger data base and an environment closer to operational use should, however, be considered to validate the projections of the study reported here.